ROBUST ADAPTATION OF NATURAL LANGUAGE PROCESSING FOR LANGUAGE VARIATION

A Dissertation
Presented to
The Academic Faculty

By

Yi Yang

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy in the
School of School of Interactive Computing

Georgia Institute of Technology

May 2017

ROBUST ADAPTATION OF NATURAL LANGUAGE PROCESSING FOR LANGUAGE VARIATION

Approved by:

Professor Jacob Eisenstein, Advisor
School of Interactive Computing
Georgia Institute of Technology

Professor James Rehg
School of Interactive Computing
Georgia Institute of Technology

Professor Byron Boots
School of Interactive Computing
Georgia Institute of Technology

Professor Polo Chau
School of Computational Science and Engineering
Georgia Institute of Technology

Professor Hal Daumé III
Department of Computer Science
University of Maryland

Date Approved: November 18, 2016

To my family.

ACKNOWLEDGEMENTS

This dissertation would not have been possible without the support of many people. First, I would like to express my sincere thanks to my advisor Prof. Jacob Eisenstein, for having me as his student in the middle of my PhD program, and for his guidance and tremendous support since then. The past four years with Jacob and the Georgia Tech Computational Linguistics Lab were truly incredible. Jacob taught me how to do research in natural language processing with regards to every single aspect, including coding complex models, writing decent papers, presenting work under different circumstances, and most importantly thinking ideas in a big picture. He has demonstrated what it means to be a great advisor. I am also thankful to James Rehg, Byron Boots, Polo Chau, and Hal Daumé III for being on my committee and their valuable comments that help to improve this dissertation.

The work presented in the dissertation also benefits from discussions with the student members and alumni in the Computational Linguistics Group. I learned a lot about graphical models from Yangfeng Ji, and chatted with him was often a good way to dismiss spare time of a day. I admire Umashanthi Pavalanathan for her passion in research—she is usually the first one came to the Lab and the last one left. Parminder Bhatia, Naman Goyal, and especially Ana Smith read my paper drafts and helped to improve my English writing. Thanks also go to Sandeep Soni, Ian Stewart, Yuval Pinter, Dong Nguyen, Vinodh Krishnan, Gongbo Zhang, Akanksha, Rahul Goel, Karishma Malkan, Patrick Violette, Yijie Wang, and Robert Guthrie. I also want to thank my friends, Yin Li, Xin Zhang, Jige Quan, and Chen Tang for sharing opinions about computer science and general AI and for spending exciting and boring days in Atlanta together.

I am lucky to have worked with Ming-Wei Chang, Scott Wen-tau Yih, Chris Meek, Bjoern Hoffmeister, Imre Kiss, and Lambert Mathias during my internship at Microsoft Research and Amazon. I benefited greatly from discussions with them, for learning more

about industry research and for shaping my research interests. Ming-Wei is like a friend to me, and I learned a lot about structured prediction, SVM, and information extraction from him. Scott is a mentor model, who has great insights about research frontiers and who is always supportive for his interns.

Finally, I would like to thank my family for their love and support during the long journey across an ocean. Thank you Mom for making me who I am now. Thank you Chenchen for your sacrifice and always being on my side. No words can describe how much I appreciate and love you.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

SUMMARY

Natural language processing (NLP) technology has been applied in various domains, ranging from social media and digital humanities to public health. Unfortunately, the adoption of existing NLP techniques in these areas often experiences unsatisfactory performance. Languages of new datasets and settings can be significantly different from standard NLP training corpora, and modern NLP techniques are usually vulnerable to variation in non-standard languages, in terms of the lexicon, syntax, and semantics. Previous approaches toward this problem suffer from three major weaknesses. First, they often employ supervised methods that require expensive annotations and easily become outdated with respect to the dynamic nature of languages. Second, they usually fail to leverage the valuable metadata associated with the target languages of these areas. Third, they treat language as uniform and ignore the differences in language use with respect to different individuals.

In this thesis, we propose several novel techniques to overcome these weaknesses and build NLP systems that are robust to language variation. These approaches are driven by co-occurrence statistics as well as rich metadata without the need of costly annotations, and can easily adapt to new settings. First, we can transform lexical variation into text that better matches standard datasets. We present a unified unsupervised statistical model for text normalization. The relationship between standard and non-standard tokens is characterized by a log-linear model, permitting arbitrary features. Text normalization focuses on tackling variation in lexicons, and therefore improving underlying NLP tasks. Second, we can overcome language variation by adapting standard NLP tools to fit the text with variation directly. We propose a novel but simple feature embedding approach to learn joint feature representations for domain adaptation, by exploiting the feature template structure commonly used in NLP problems. We also show how to incorporate metadata attributes into feature embeddings, which helps to learn distill the domain-invariant properties of each feature over multiple related domains. Domain adaptation is able to deal with a full range

of linguistic phenomenon, thus it often yields better performances than text normalization. Finally, a subtle challenge posed by variation is that language is not uniformly distributed among individuals, while traditional NLP systems usually treat texts from different authors the same. Both text normalization and domain adaptation follow the standard NLP settings and fail to handle this problem. We propose to address the difficulty by exploiting the sociological theory of *homophily*—the tendency of socially linked individuals to behave similarly—to build models that account for language variation on an individual or a social community level. We investigate both *label homophily* and *linguistic homophily* to build socially adapted information extraction and sentiment analysis systems. Our work delivers state-of-the-art NLP systems for social media and historical texts on various standard benchmark datasets.

CHAPTER 1
INTRODUCTION

With the rise of the Internet, huge amounts of texts of various domains are now available in digital form. For example, more than five hundred million Twitter messages are generated by users every day; more than 3.5 billion search queries are issued on Google on a daily basis; and millions of historical books have been digitized over the past decades [1]. After the steady improvements in core natural language processing (NLP) tasks like part-of-speech (POS) tagging, information extraction, and syntactic parsing with classic corpora (e.g., the *Wall Street Journal* and the *New York Times*), potential users are interested in applying these techniques to a broader range of new datasets and settings, including social media (e.g., Twitter and Facebook) and historical texts. However, these texts differ from standard NLP training corpora in a number of linguistic respects, including the lexicon [2, 3, 4], morphology [5], orthography [6], and syntax [7, 8], which is known as language variation.

Variation in the non-standard languages imposes significant challenges to modern NLP tools. For example, the accuracy of the CLAWS part-of-speech Tagger [9] drops from 97% on the British National Corpus to 82% on Early Modern English texts [10]. In named entity recognition (NER), the F1 score of Stanford recognizer that is trained with CoNLL corpora (news text) falls from 86% on the CoNLL test set [11] to 44% F1 score [12] for Twitter. Potential NLP users like social scientists and humanities researchers are clamoring to adapt language technology to social media and historical literature for obtaining new insights on public health [13, 14], political attitudes [15, 16], products and businesses [17], and literary scholarship [18], but the problems at the core of the NLP pipeline greatly limit the performance of these downstream applications.

In the thesis, we propose to explore several novel techniques to address the problems

posed by variation, building NLP systems that are more robust to language variation. **Text normalization** copes with linguistic variation by transforming lexical variants into their standard forms that better match standard datasets. It has been shown to yield improvements for NLP tasks such as part-of-speech tagging and machine translation [19, 20]. Instead of changing the input text, **domain adaptation** works with specific tasks and adapts standard NLP systems to fit the text with variation directly. Compared with text normalization, domain adaptation can handle a full range of linguistic aspects, which often leads to better performance for particular tasks. However, domain adaptation usually focuses on a single downstream application, while text normalization can be applied as a preprocessing step and improve multiple NLP tasks. Both text normalization and domain adaptation attempt to make non-standard datasets resemble standard training corpora and eliminate differences between non-standard languages and standard languages. We can improve NLP systems even further by accounting for individual-level linguistic variation. However, it is impractical to obtain annotated data for each individual user. **Socially adapted natural language processing** tackles this problem and overcomes language variation by leveraging social network information, as variation is usually linked to social factors that can be characterized by social structures.

The approaches studied in the thesis are largely complementary, and aim at addressing different aspects of variation in languages and applying to different application scenarios. Domain adaptation and socially adapted NLP require joint learning for specific downstream tasks, which can be quite complicated for high-level NLP applications like question answering and machine translation; while text normalization can be treated as a simple preprocessing step that often works well for these tasks. On the other hand, because of the joint training, domain adaptation and socially adapted NLP can handle different types of linguistic variation. Therefore, they usually perform better than text normalization on a variety of NLP tasks such as part-of-speech tagging, named entity recognition, and sentiment analysis. Finally, as shown in Figure 1.1, text normalization requires to predefine

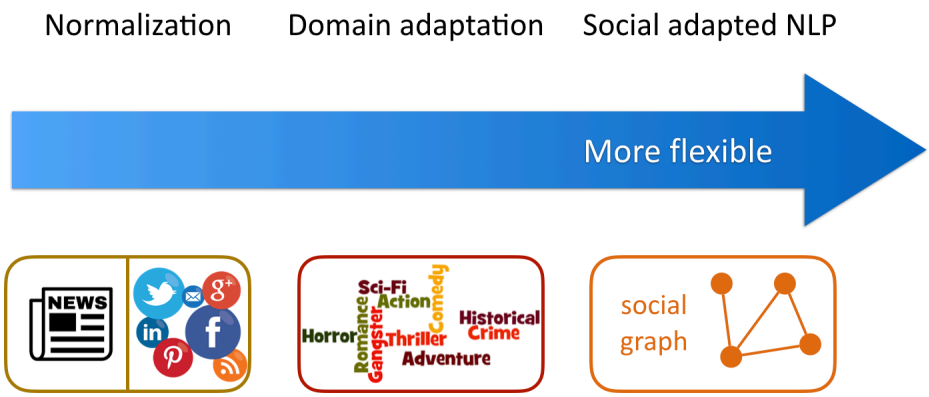
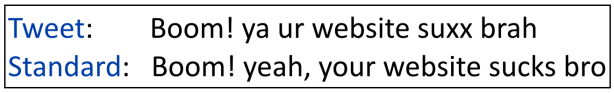


Figure 1.1: Generalizability of different approaches. Text normalization requires to predefine the specific target language; domain adaptation can work with a few metadata attributes such as genres and temporal epochs; and socially adapted NLP only asks for a simple social network.

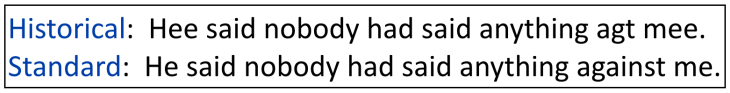
the specific target language, like news texts. Although domain adaptation also needs to distinguish the source domain and the target domain, sometimes we may be able to relax the requirement to a few metadata attributes such as genres and temporal epochs, making it generalizable to more application scenarios. Socially adapted NLP is the most flexible approach, as it only asks for a simple social network, which is available in a wide range of contexts, from social media [21] to political speech [22] to historical texts [23]. Overall, the proposed methods exploit distributional semantics and rich metadata associated with the texts, generalizing NLP systems to better process non-standard languages. The goal of this thesis is to deliver reliable and effective NLP tools to a wider range of domains, including social media, digital humanities and public health.

1.1 Unsupervised Text Normalization

Text normalization involves mapping lexical variants in non-standard languages to their canonical forms in standard languages, thus bridging the gap between standard training corpora and target non-standard texts. Figure 1.2 shows two sentences from non-standard languages and their normalizations. This task is non-trivial, as it is surprisingly difficult to find a precise definition for the normalization task. For example, abbreviations like 'lol' and



a) Twitter message



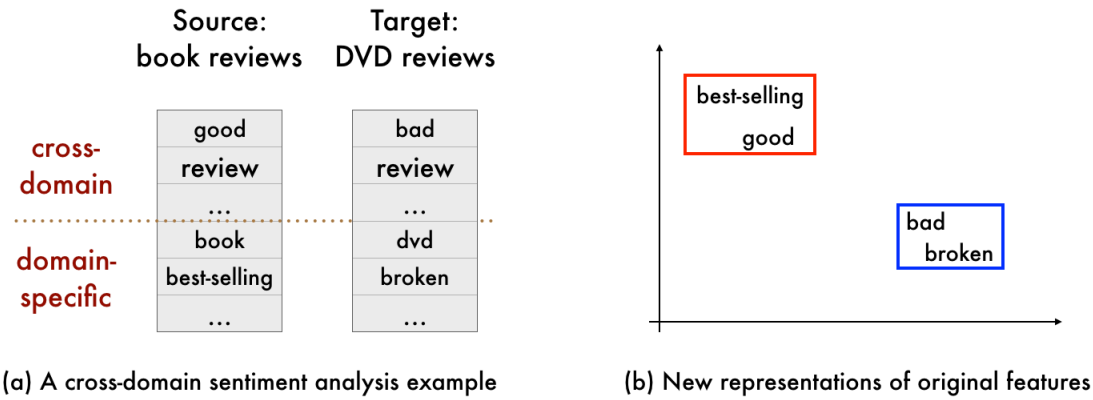
b) Early Modern English sentence

Figure 1.2: Two example sentences from non-standard languages and their normalizations. The tweet was posted by Sarah Silverman, and the Early Modern English sentence was written by Henry Oxinden (1660).

'wtf' are usually not normalized in existing work, even when they are used to abbreviate syntactic constituents, as in 'wtf is the matter with you?'. The non-standard token 'smh' can be normalized to 'somehow' or 'shake my head', depending on the context. In order to define a treatable task, Han and Baldwin [24] come out with a dataset with a focus on a subset of the normalization problem—annotating only token-to-token normalizations.

We want to be able to build an automatic normalization system without requiring labeled data. At present, labeled data for text normalization is available only in small quantities. Moreover, as social media language is undergoing rapid change [25], labeled datasets may become stale and increasingly ill-suited to new spellings and words. There are two main sources of information to be exploited for unsupervised text normalization: local context, and surface similarity between the observed strings and normalization candidates. In Figure 1.2(b), we consider 'He' as the normalization output for 'Hee', as the Levenshtein distance between 'Hee' and 'He' is one, and they are also frequently followed by similar context words like 'said' and 'told'. In practice, additional external resources such as slang dictionaries and spell checkers are often utilized to further improve performance of normalization systems [26, 24, 19].

In chapter 3, we present an unsupervised statistical model for text normalization. Similar to existing systems, our model takes advantage of both surface similarity and local context. However, prior work usually employs pipeline architectures, while our approach is a unified model that permits arbitrary features. Our normalization system achieves the



(a) A cross-domain sentiment analysis example (b) New representations of original features

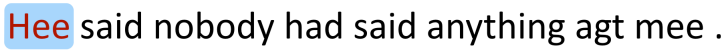
Figure 1.3: Illustration of unsupervised domain adaptation for sentiment analysis. (a) The two domain-specific features 'best-selling' and 'broken' tend to co-occur with the cross-domain features 'good' and 'bad' respectively. (b) 'best-selling' and 'broken' will be far away from each other under the new representations, therefore they should express opposite semantic meanings as those of 'good' and 'bad'.

best results on a standard Twitter normalization benchmark without requiring additional resources. We also introduce a new training approach for unsupervised learning problems with large label spaces.

1.2 Unsupervised Multi-Domain Adaptation

Although text normalization has been shown to yield improvements for some NLP tasks [20, 27], it has been criticized by Eisenstein [25], who argues that it is not always a well-defined problem, and it strips away important social meanings (as shown in Figure 1.2(a)). In addition, current state-of-the-art normalization tools often yield unsatisfactory performance. For example, according to the VARD historical spelling normalization tool [28], 'agt' in Figure 1.2(b) is incorrectly normalized to 'aged', due to the common subword mapping rule '-t $\rightarrow$ -ed' between historical English and contemporary English. Domain adaptation aims at adapting NLP tools to fit the variation in languages directly. This process avoids the change of raw inputs, and often results in better performance than text normalization for downstream tasks.

Specifically, unsupervised domain adaptation is more appealing than its supervised



Feature template	Feature value
Current word	hee
Next word	said
Prefix 1	h
Suffix 1	e

Figure 1.4: Examples of feature templates and corresponding feature values for part-of-speech tagging task. We are interested in tagging the token "Hee" in the example sentence.

counterpart, since it requires no labeled data in the target domain. The typical approach for the problem is learning joint feature representations on the union of the training and target data, so that the source and target instances would be more similar under the new representations. Feature co-occurrence statistics are the primary source of information driving these unsupervised methods. Consider the example presented in Figure 1.3. The main difficulty with this cross-domain sentiment analysis task—training a sentiment classifier on book reviews and testing on DVD reviews—is that the target domain data contains many out-of-vocabulary (OOV) features, which never appear in the source domain data (e.g., 'dvd', 'broken'). The classifier trained on the source data is not able to learn corresponding weights for these OOV features. The OOV features are also referred as target-specific features. On the other hand, there are source-specific features that never appear in the target data, so that the learned weights of these features are useless for the cross-domain sentiment classification task. In order to address this problem, as shown in Figure 1.3, we can transform the original features into low-dimensional representations using the feature co-occurrence statistics. The basic idea is that if one source-specific feature and one target-specific feature tend to co-occur with similar cross-domain features (i.e., features appears in both source and target data), the two domain-specific features should be semantically similar to each other. The similarity can be measured under the new representations trained with techniques such as autoencoders, and singular value decomposition.

We propose a novel but simple feature embedding approach for unsupervised domain

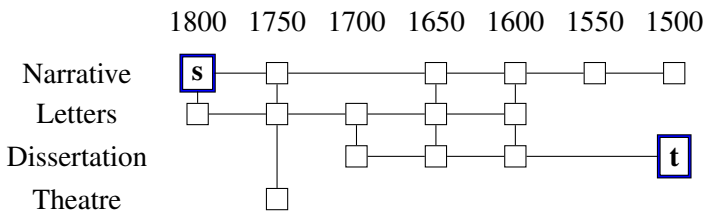


Figure 1.5: Domain graph for the Tycho Brahe corpus [29]. **Suppose we want to adapt from 19th Century narratives to 16th Century dissertations: can unlabeled data from other domains help?**

adaptation. First, most prior work relies on the selection of *pivot features* that requires some task-specific heuristics (e.g., the most frequent cross-domain features). Our method avoids this procedure by exploiting the feature template structure common in NLP problems. Figure 1.4 show some feature templates and the corresponding feature values for the task of part-of-speech tagging. As shown, there is exactly one active feature for every feature template given a token instance. We use this template structure to induce feature embeddings, so that each embedding is selected to help predict the features that fill out the other templates.

Second, unsupervised domain adaptation is typically treated as a task of moving from a single source to a single target domain. In reality, test data may be diverse, relating to the training data in some ways but not others. For example, as shown in Figure 1.5, we may be given part-of-speech labeled text from 19th Century narratives, and we hope to adapt the tagger to work on academic dissertations from the 16th Century. The unlabeled data from other domains can be also relevant and useful. We extend the feature embedding idea to unsupervised multi-attribute domain adaptation, where each domain is characterized by some domain attributes, such as genres (e.g., Narrative, Letters) and temporal epochs. For each feature, our model learns both attribute-independent and attribute-specific feature embeddings. The attribute-independent feature embeddings capture domain-neutral information, and therefore be more robust to domain shift. The details will be discussed in chapter 4.

1.3 Socially Adapted Natural Language Processing

People use language in different ways, influenced by extra-linguistic factors such as age, gender, race, geography, and more ineffable characteristics such as political and cultural attitudes. Most NLP tasks, however, treat language as uniform, and ignore the fact that whether a text was produced by a middle-aged man, an elderly lady, or a teenager. The simplification can harm performance of NLP systems, as these three groups differ along a whole host of demographic axes, which are reflected in their language use. Domain adaptation is relevant to this problem. For example, Hovy [30] employs supervised domain adaptation techniques to improve the accuracy of sentiment analysis and topic classification by the inclusion of coarse-grained author demographics such as age and gender. However, such demographic information is not directly available in most datasets, and it is not yet clear whether predicted age and gender offers any improvements. On the other end of the spectrum are attempts to create *personalized* language technologies, as are often employed in information retrieval [31], recommender systems [32], and language modeling [33]. But personalization requires annotated data for each individual user—something that may be possible in interactive settings such as information retrieval, but is not typically feasible in natural language processing.

We propose a middle ground between group-level demographic characteristics and personalization, by exploiting social network structure. The sociological theory of *homophily* asserts that individuals are usually similar to their friends [34]. This property has been demonstrated both for language [35, 36] as well as for the demographic properties targeted by Hovy [30], which are more likely to be shared by friends than by random pairs of individuals [37, 38]. Social network information is available in a wide range of contexts, from social media [21] to political speech [22] to historical texts [23]. Thus, social network homophily has the potential to provide a general and effective way to account for linguistic variation in NLP. Our goal is to build socially adapted NLP systems. In particular, the

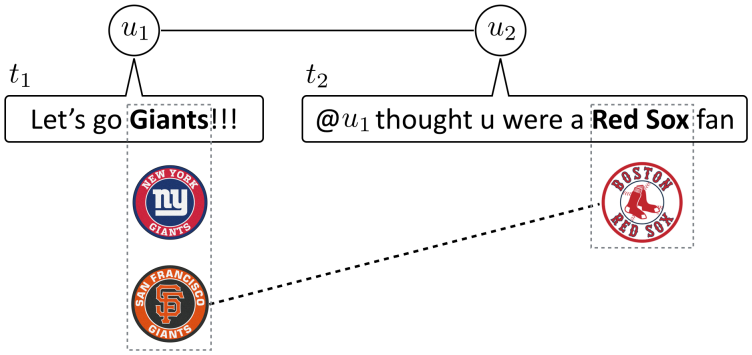


Figure 1.6: Illustration on leveraging social relations for entity disambiguation. Socially connected users u_1 and u_2 tend to talk about similar entities (baseball in the example).

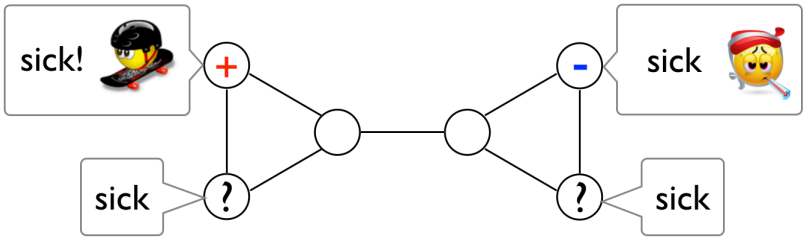


Figure 1.7: Words such as 'sick' can express opposite sentiment polarities depending on the user. We account for this variation by generalizing across the social network.

classification function for a specific user depends on functions for its social neighbors.

In chapter 5, we present two socially adapted NLP systems for entity linking and sentiment analysis respectively. Our tweet entity linking system is based on the assumption of *entity homophily*—Twitter users will have similar interests in real world entities to their near neighbors—which is demonstrated in Figure 1.6. The social relation between users u_1 and u_2 may lead to more coherent topics in tweets t_1 and t_2. Therefore, by successfully linking the less ambiguous mention 'Red Sox' in tweet t_2 to the *Boston Red Sox* baseball team, the tweet entity linking system will be more confident on linking 'Giants' to the *San Francisco Giants* baseball team in tweet t_1. Entity homophily operates on the task labels directly, which may be not appropriate for some other NLP problems. Our sentiment analysis system takes a step further and makes the *linguistic homophily* assumption. As illustrated in Figure 1.7, given labeled examples of 'sick' in use by individuals in a social network, we assume that the word will have a similar sentiment meaning for their near neighbors.

Note that this differs from the assumption of *label homophily*, which entails that neighbors in the network will hold similar opinions, and will therefore produce similar document-level labels [22, 39, 40]. Linguistic homophily is a more generalizable claim, which could in principle be applied to any language processing task where author network information is available. Our systems incorporate distributed author representations trained with social network structures into neural network models, achieving significant improvements on benchmark datasets for entity linking and sentiment analysis.

1.4 Outline and Contributions

After discussing some background knowledge about language variation and change in chapter 2, we present the main bulk of the thesis in the following chapters. The contributions of this thesis are:

- A log-linear model, performing unsupervised text normalization in a maximum-likelihood framework. It allows to capture the relationship between standard and non-standard tokens with arbitrary features (chapter 3). This chapter is largely based on work previously published by Yang and Eisenstein [41].

- A novel representation learning approach, feature embedding, for unsupervised domain adaptation (chapter 4). The method is then extended for another related task—unsupervised multi-domain adaptation. This chapter is largely based on work previously published by Yang and Eisenstein [42] and Yang and Eisenstein [43].

- Two socially adapted models for tweet entity linking and sentiment analysis, leveraging the assumptions of entity homophily and linguistic homophily respectively (chapter 5). This chapter is largely based on work previously published by Yang *et al.* [44] and Yang and Eisenstein [45].

1.5 Thesis Statement

In this thesis, we advocate for computational methods that can better cope with variation across several domains of non-standard text data. People generate texts in different ways, depending on the characteristics of the authors, as well as the communication media. We argue that building NLP systems that account for language variation will significantly improve multiple NLP tasks, and benefit various high impact application areas such as social media, patient medical records, and historical texts. We demonstrate that co-occurrence statistics and metadata such as social network structure can help to induce robust representations with respect to variation. The resulting systems have turned out to work well for a range of text analysis tasks, including part-of-speech tagging, named entity recognition, tweet entity linking, and sentiment analysis.

CHAPTER 2

BACKGROUND

2.1 Language Variation and Language Change

Variation in language is pervasive, as language usage varies from group to group, and speaker to speaker, in terms of the pronunciation of a language (phonological variation), the choice of words (lexical variation) and the meaning of those words (semantic variation), and even the use of syntactic constructions (syntactic variation). A well-known example is that the English usage of Americans is noticeably different from that of British. In United States, the language usage of African Americans is also notably different from that of other groups of people. They are different *dialects* of English. Consider the following sentence from Dillard [46]:

'You makin' sense, but you don' be makin' sense!'

This is a sentence of African American English (AAE), and speakers of the standard dialect of English are likely to conclude that it is ungrammatical. The first clause lacks a verb (such as 'are') that the standard dialect requires, and the second clause contains the 'do + be' sequence that the standard dialect prohibits. In the extreme limit, no two speakers of a language produce and use their language in exactly the same way. The form of a language spoken by a single individual is defined as an *idiolect*.

From a linguistic perspective, no one dialect of a language is any more correct or any better than the other dialects. However, governments and societies in the contemporary world need to design one dialect of a language as the standard form of the language, for the sake of legal and governmental functions, as well as educational purpose. In the contemporary United States, Standard American English (SAE) is a form of the language used in news programs in the national media, which is perhaps most widely identified with the

educated white middle class. Similarly, in countries throughout the world, the standard language is usually the dialect of the subculture with the most prestige and power.

Besides the aforementioned dialects, variation is a common characteristic of non-standard languages of various domains, such as biomedical documents, social media texts, and historical texts. Physicians and health professionals use medical jargon, a special or technical vocabulary, in oral or written communication. The particular jargon, which is not intended to be secret, but, for practical reasons, are largely incomprehensible to those outside the particular profession or group that uses the jargon. The informal nature of social media communication brings a lot of slang terms and lexical variants. As shown in Figure 1.2, social network users also intend to express special personalities or social meanings via the use of language in some non-standard styles. Finally, human language is undergoing evolution, and small changes are made from time to time. Historical languages may be still largely intelligible to contemporary speakers, but they differ from modern languages in a number of linguistic respects, including the lexicon [4], morphology [5], and syntax [8].

2.1.1 Historical texts

Languages are naturally undergoing change over time. Most of the modern European languages, such as English, German, and French, were historically related to each other, and they were also related to other ancient languages, such as Latin, Greek, and Sanskrit. It turns out that they all belong to the Indo-European Language family. From time to time, the speakers diversified the languages by developing specific linguistic rules for different languages, with respect to phonology, lexicon, morphology, semantics, and syntax. Earlier changes in languages involve the improvement in expressive ability, and recent changes aim at maintaining a balance in expressiveness and grammatical complexity over time.

Linguists traditionally distinguish three major periods for the development of English language, though the language change were continuous and gradual in general. The Old English period begins at fifth century and ends in eleventh century; the Middle English period

is from eleventh century to fifteenth century; and the Modern English period starts from fifteenth century until present. The English language has undergone extensive changes between the Old and Modern English periods, due to the Norman Conquest of England by William the Conqueror in 1066. The Normans brought England people the French language, and the English language is significantly influenced by French since then.

Figure 1.2 has shown a sentence of Early Modern English. We now examine a few systematic changes between historical English and Modern English. During the development of Modern English, a great number of words were brought from other languages. For example, the words 'pork', 'beef', 'veal', 'mutton', and 'venison' are derived from French words referring respectively to the edible meat of the 'swine', 'cow', 'calf', 'sheep', and 'deer'. There are examples of individual words undergoing meaning changes. A well-known example is the word 'hot', which was solely described physical temperature, undergoes a metaphorical extension, as used in 'hot news' (breaking news) and 'hot car' (stolen car). Language change in syntax includes the Particle Movement rule—a new added syntactic rule to English since the Old English period. One application of the rule is the sentence pair: 'John threw out the fish' and 'John threw the fish out'.

2.1.2 Social media texts

Over the past decade, online social networking sites, such as Facebook, Twitter, Google+, and LinkedIn, have revolutionized the way we communicate with individuals, groups, and communities, and has changed everyday practices. On most of the social media platforms, people generate text content for different proposes: exchanging information and messages (e.g., Facebook, Twitter); posting specialized information, questions, or answers (e.g., StackOverflow, CNET forums, Apple Support); sharing information and opinions (e.g., WordPress, Mashable, Boing Boing). As of July 2015, there are 2.3 billion active social media users among the 3.17 billion Internet users. On WordPress alone, 56 million blog posts are published every month. There are 500 million Twitter messages are produced

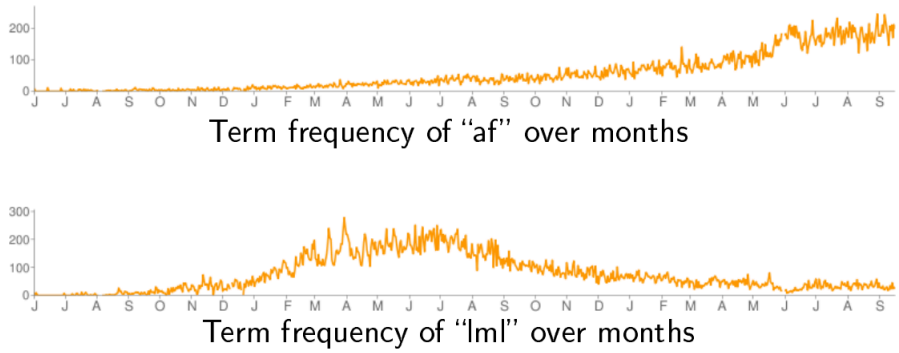


Figure 2.1: Illustration of the change of Twitter language. The term frequencies are obtained from a corpus with one million English-language tweets [47].

each day, corresponding to 6,000 tweets every second.[1]

The emerging of social media texts poses significant challenge on NLP systems. It introduces a large amount of variation in the languages, since the texts are usually produced in informal, conversational environments. Compared with historical texts, the variation in social media texts is less systematic—there are non-standard spellings, but also a lot of noise and limited sets of textual features. Consider this tweet: '#qcpoli enjoyed a hearty laugh today with #plq debate audience for @jflisee #notrehome tune was that the intended reaction?' Inconsistent (or absent) punctuation and capitalization can make detection of sentence boundaries quite difficult. Novel features like hashtag and user mention, incorrect or non-standard spelling, and rampant abbreviations complicate tokenization. The abundance of user-generated content also comes at a price: there may be high-quality content, but also much spam content such as advertisements, self-promotion, pointless babbles, or misleading information [48]. Moreover, many social network sites impose a set of constraints on the content (e.g., the 140-character limit for Twitter posts), making context-aware tasks like named entity recognition and event detection troublesome. Finally, social media languages experience rapid changes over a short time period. As shown in Figure 2.1, the popularities of the two terms have undergone dramatic changes over two years.

On the other hand, social media texts are often associated with rich metadata informa-

[1] https://www.brandwatch.com/2016/03/96-amazing-social-media-statistics-and-facts-for-2016/

tion. First, social network information is available in most social media sites: individuals are able to befriend (e.g., Facebook) or follow (e.g., Twitter) each other, forming social connections. We may induce social networks from metadata attributes, such as Twitter retweet and mention relations directly. Second, the rise of social networks has made everybody a potential author, so the language is now closer to the individual user than to any prescribed norms. Thus, the user profile information that is available in many social media platforms become fairly relevant if we want to build better text analysis systems by accounting for language variation. As mentioned above, variation in languages is linked to demographics, such as age [49], gender [50], race [51], and geography [52]. Finally, metadata can play an important role in obtaining annotated training data for supervised machine learning systems. Some metadata attributes, such as review ratings, and content tags, offer valuable label information for supervised learning tasks, and help to avoid expensive data annotation.

2.2 Adaptation of NLP Systems

The availability of non-standard text data enables people survey and design numerous novel applications that have drawn considerable public attention. Sentiment analysis on the texts is a popular application that helps to obtain new insights in the healthcare, financial, news media domains. Ali *et al.* [53] collected texts from medical forums about hearing devices and classified them into negative, neutral, or positive classes. Economic indicators are usually computed via traditional socio-economic surveys. Mao and Bollen [54] investigated whether the results of a variety of such surveys can be replicated by performing sentient analysis on Twitter data. Information extraction is another important task underlying many downstream applications. Nagarajan *et al.* [55] used Twitter to extract crowd-sourced observations on spatio-temporal-thematic analysis to real-world events. Shamma *et al.* [56] studied the live visual media events using conversational activity on video footage and Twitter during two events of the first 2008 U.S. Presidential Debate and the Inauguration

of Barack Obama.

Nevertheless, the success of these downstream studies highly relies on accurate core NLP tools for non-standard texts. Existing NLP techniques are driven by standard corpora, such as the Penn Treebank [57] and the CoNLL corpus for named entity recognition [58]. The state-of-the-art NLP methods applied to non-standard texts are therefore confronted with difficulties due to non-standard spelling, noise, and limited sets of features for automatic clustering and classification. The automatic processing of social media or historical data needs to design appropriate research methods for core NLP tasks such as part-of-speech (POS) tagging, dependency parsing, chunking, and named entity recognition (NER). There have been a few attempts to address the problems posed by variation for the accuracy of NLP systems, primarily branching into three directions: obtaining annotation data for non-standard domains, normalizing non-standard spellings into their canonical forms, and adapting existing NLP systems via domain adaptation techniques.

2.2.1 Data annotation

The most straightforward approach is obtaining new annotated training data of different NLP pipelines, and re-train the models for non-standard texts. In general, it is easy to collect a large amount of non-standard text, but it is difficult and time-consuming to annotate the text. Currently, some annotated social media data has become available, but the volume is not high enough. Several NLP tools have been re-trained on newly annotated data, and some approaches are also able to leverage some unannotated text in addition to the small amounts of annotated text.

Gimpel *et al.* [59] developed a POS tagger for Twitter, which utilizes a new tagset that is more coarse-grained compared with the Penn Treebank tagset. The tagset pays particular attention to Twitter-specific characteristics, such as emoticons, mentions, and hashtags. They manually tagged 1,827 tweets with the new tagset. Then, they trained a POS tagging model with some features designed for Twitter text. The experiment results showed that

the model achieves 90% accuracy for the POS tagging task. Owoputi *et al.* [60] improved on the model by using word clustering techniques and trained the POS tagger on a better dataset of tweets and chat messages.

Kong *et al.* [61] built a dependency parser named TweeboParser for tweets. It is developed specifically on a recently annotated Twitter treebank for 929 tweets, which consists of a subset of tweets from Gimpel *et al.* [59]. It also uses the same POS tagset as used in Gimpel *et al.* [59]. The authors performed an extensive survey of key challenges posed by syntactic analysis of tweets and decisions motivated by those challenges and by the limited annotation-resource scenario. Then, they adapted an existing statistical parsing algorithm with several novel features for the non-standard domain. They also proposed an approach to exploit out-of-domain Penn Treebank data. The experiments showed that the parser obtained over 80% unlabeled attachment accuracy on the test set.

Ritter *et al.* [12] re-built the NLP pipeline for Twitter messages beginning with part-of-speech tagging, through chunking, to named-entity recognition. The work focused on the NER task, on which the authors introduced a novel approach to distant supervision using Topic Models. In particular, LabeledLDA [62] was applied, utilizing constraints based on an open-domain database (Freebase) as a source of supervision. Their best NER system reached an F1 score of 67%, which doubles F1 score compared with the Stanford NER system.

2.2.2 Spelling normalization

Spelling normalization is another solution for overcoming language variation. The task can be approached in two stages: first, identifying orthographic non-standard spellings in an input text, and second, mapping lexical variants to their canonical forms in standard languages. The text normalization task was introduced by Sproat *et al.* [63], and attained popularity in the context of SMS messages [64]. It has become more salient in the context of social media texts and historical texts. The popular spelling normalization tool

for historical English, VARD [28], takes a straightforward absolute approach to spelling canonicalization. It uses special dictionaries that associate observed historical variants with modern word forms; the mapping is precise and can handle cases where the association is non-trivia or when no canonical cognate exists. Han *et al.* [24] formally defined a normalization task for Twitter, focusing on normalizations between single tokens, and excluding multi-word tokens like 'lol' ('laugh out loud'). In recent work, normalization has been shown to yield improvements for part-of-speech tagging [10, 19], parsing [27], and machine translation [20].

Early work on normalization focused on labeled SMS datasets, using approaches such as noisy-channel modeling [64] and machine translation [65], as well as hybrid combinations of spelling correction and speech recognition [66, 67]. Cook *et al.* [68] manually identified several word formation types within a noisy channel framework. They parametrized each formation type with a small number of scalar values, so that all legal transformations of a given type are equally likely. The scalar parameters are then estimated using expectation maximization. Contractor *et al.* [69] used string edit distance to identify closely-related candidate orthographic forms and then decoded the message using a language model. Gouws *et al.* [70] refined this approach by mining an "exception dictionary" of strongly-associated word pairs such as 'you/u'.

Recent approaches have sought to improve accuracy by bringing more external resources and complex architectures to bear. Han *et al.* [24] began with a set of string similarity metrics, and then applied dependency parsing to identify contextually-similar words. Liu *et al.* [26] extracted noisy training pairs from the search snippets that result from carefully designed queries to Google, and then trained a conditional random field [71] to estimate a character-based translation model. They later extended this work by adding a model of visual priming, an off-the-shelf spell-checker, and local context [72]. Hassan and Menezes [20] used a random walk framework to capture contextual similarity, which they then interpolated with an edit distance metric.

2.2.3 Domain adaptation

Instead of changing the raw input via text normalization, we may employ domain adaptation techniques to fit the variation in languages directly, as it is natural to think of one non-standard language as a distinct domain from the standard news texts. Domain adaptation has been a popular topic in machine learning literature over the past few years, and the main challenge is that the data distribution of our training set (i.e., source domain) is different from that of the test set (i.e., target domain). People have been worked on overcoming the mismatch in the data distributions using both supervised and unsupervised methods.

Supervised domain adaptation assumes that we have a small amount of labeled data in the target domain. Typical supervised domain adaptation approaches addressed the data distribution mismatch problem by projecting source and target instances into similar spaces, focusing on adapting the classifier weights across domains. Daumé III [73] proposed a kernel-mapping function that maps the data from both source and target domains to a high-dimensional feature space, where standard discriminative learning methods are used to train the classifiers. Finkel and Manning [74] propagated classification parameters across a tree of domains using Bayesian priors, so that classifiers for sibling domains are more similar.

Recent work has been focused on unsupervised domain adaptation, where no labeled data is available in the target domain. Several representation learning methods have been proposed to solve this problem, as representational differences between source and target domains can be a major source of errors in the target domain [75]. Cross-domain representations were first induced via auxiliary prediction problems [76], such as the prediction of *pivot features* [77]. In these approaches, as well as in later work on denoising autoencoders [78], the key mechanism is to learn a function to predict a subset of features for each instance, based on other features of the instance. An alternative approach for unsupervised domain adaptation is to link unsupervised learning in the source and target domains with the label distribution in the source domain, through the framework of posterior regularization [79, 80, 81].

CHAPTER 3

A LOG-LINEAR MODEL FOR UNSUPERVISED TEXT NORMALIZATION

In this chapter we introduce a novel log-linear model for unsupervised text normalization, focusing on normalizing social media posts into standard sentences that resemble news texts. As discussed in chapter 1, supervised training is generally not considered appropriate for text normalization. However, due to the extremely high-dimensional output space—arbitrary sequences of words across the vocabulary—it is a very challenging problem for unsupervised learning. Perhaps it is for these reasons that the most successful systems are pipeline architectures that cobble together a diverse array of techniques and resources, including statistical language models, dependency parsers, string edit distances, off-the-shelf spellcheckers, and curated slang dictionaries [26, 24, 19].

We propose a different approach, performing normalization in a maximum-likelihood framework. We treat the local context using standard language modeling techniques; we treat string similarity with a log-linear model that includes features for both surface similarity and word-word pairs. As the label space for the task is huge, which is the standard vocabulary itself, with at least 10^4 elements, the standard training techniques such as dynamic programming are prohibited. We propose a new training approach to overcome the challenge, using Monte Carlo techniques to compute an approximate gradient on the feature weights.

This model is implemented in a normalization system called UNLOL (**u**nsupervised **n**ormalization in a **LO**g-**L**inear model). It is a lightweight probabilistic approach, relying only on a language model for the target domain; it can be adapted to new corpora text or new domains easily and quickly. Our evaluations show that UNLOL outperforms the state-of-the-art on standard normalization datasets. In addition, we demonstrate the linguistic insights that can be obtained from normalization, using UNLOL to identify classes

of orthographic transformations that form coherent linguistic styles.

3.1 Approach

Our approach is motivated by the following criteria:

- **Unsupervised**. We want to be able to train a model without labeled data. At present, labeled data for Twitter normalization is available only in small quantities. Moreover, as social media language is undergoing rapid change [25], labeled datasets may become stale and increasingly ill-suited to new spellings and words.

- **Low-resource**. Other unsupervised approaches take advantage of resources such as slang dictionaries and spell checkers [24, 26]. Resources that characterize the current state of internet language risk becoming outdated; in this paper we investigate whether high-quality normalization is possible without any such resources.

- **Featurized**. The relationship between any pair of words can be characterized in a number of different ways, ranging from simple character-level rules (e.g., *going*/`goin`) to larger substitutions (e.g., *someone*/`sum1`), and even to patterns that are lexically restricted (e.g., *you*/`u`, *to*/`2`). For these reasons, we seek a model that permits many overlapping features to describe candidate word pairs. These features may include simple string edit distance metrics, as well as lexical features that memorize specific pairs of standard and nonstandard words.

- **Context-driven**. Learning potentially arbitrary word-to-word transformations without supervision would be impossible without the strong additional cue of local context. For example, in the phrase

  `give me suttin to believe in`,

  even a reader who has never before seen the word `suttin` may recognize it as a phonetic transcription of *something*. The relatively high string edit distance is overcome

by the strong contextual preference for the word *something* over orthographically closer alternatives such as *button* or *suiting*. We can apply an arbitrary target language model, leveraging large amounts of unlabeled data and catering to the desired linguistic characteristics of the normalized content.

- **Holistic**. While several prior approaches—such as normalization dictionaries—operate at the token level, our approach reasons over the scope of the entire message. The necessity for such holistic, joint inference and learning can be seen by changing the example above to:

  `gimme suttin 2 beleive innnn`.

  None of these tokens are standard (except `2`, which appears in a nonstandard sense here), so without joint inference, it would not be possible to use context to help normalize `suttin`. Only by jointly reasoning over the entire message can we obtain the correct normalization.

These desiderata point towards a featurized sequence model, which must be trained without labeled examples. While there is prior work on training sequence models without supervision [82, 83], there is an additional complication not faced by models for tasks such as part-of-speech tagging and named entity recognition: the potential label space of standard words is large, on the order of at least 10^4. Naive application of Viterbi decoding—which is a component of training for both Contrastive Estimation [82] and the locally-normalized sequence labeling model of Berg-Kirkpatrick *et al.* [83]—will be stymied by Viterbi's quadratic complexity in the dimension of the label space. While various pruning heuristics may be applied, we instead look to Sequential Monte Carlo (SMC), a randomized algorithm which approximates the necessary feature expectations through weighted samples.

3.2 Model

Given a set of source-language sentences $S = \{\mathbf{s}_1, \mathbf{s}_2, \ldots\}$ (e.g., Tweets), our goal is to transduce them into target-language sentences $T = \{\mathbf{t}_1, \mathbf{t}_2, \ldots\}$ (standard English). We are given a target language model $P(\mathbf{t})$, which can be estimated from some large set of unlabeled target-language sentences. We denote the vocabularies of source language and target language as ν_S and ν_T respectively.

We define a log-linear model that scores source and target strings, with the form

$$P(\mathbf{s}|\mathbf{t};\theta) \propto \exp\left(\theta^\top \mathbf{f}(\mathbf{s},\mathbf{t})\right). \tag{3.1}$$

The desired conditional probability $P(\mathbf{t}|\mathbf{s})$ can be obtained by combining this model with the target language model, $P(\mathbf{t}|\mathbf{s}) \propto P(\mathbf{s}|\mathbf{t};\theta)P(\mathbf{t})$. Since no labeled data is available, the parameters θ must be estimated by maximizing the log-likelihood of the source-language data. We define the log-likelihood $\ell_\theta(\mathbf{s})$ for a source-language sentence $\mathbf{s}$ as follows:

$$\ell_\theta(\mathbf{s}) = \log P(\mathbf{s}) = \log \sum_{\mathbf{t}} P(\mathbf{s}|\mathbf{t};\theta)P(\mathbf{t})$$

We would like to maximize this objective by making gradient-based updates.

$$\begin{aligned}
\frac{\partial \ell_\theta(\mathbf{s})}{\partial \theta} &= \frac{1}{P(\mathbf{s})} \sum_{\mathbf{t}} P(\mathbf{t}) \frac{\partial}{\partial \theta} P(\mathbf{s}|\mathbf{t};\theta) \\
&= \sum_{\mathbf{t}} P(\mathbf{t}|\mathbf{s}) \left(\mathbf{f}(\mathbf{s},\mathbf{t}) - \sum_{\mathbf{s}'} P(\mathbf{s}'|\mathbf{t}) \mathbf{f}(\mathbf{s}',\mathbf{t}) \right) \\
&= E_{\mathbf{t}|\mathbf{s}}[\mathbf{f}(\mathbf{s},\mathbf{t}) - E_{\mathbf{s}'|\mathbf{t}}[\mathbf{f}(\mathbf{s}',\mathbf{t})]]
\end{aligned} \tag{3.2}$$

We are left with a difference in expected feature counts, as is typical in log-linear models. However, unlike the supervised case, here *both* terms are expectations: the outer expectation is over all target sequences (given the observed source sequence), and the nested

expectation is over all source sequences, given the target sequence. As the space of possible target sequences $\mathbf{t}$ grows exponentially in the length of the source sequence, it will not be practical to compute this expectation directly.

Dynamic programming is the typical solution for computing feature expectations, and can be applied to sequence models when the feature function decomposes locally. There are two reasons this will not work in our case. First, while the forward-backward algorithm would enable us to compute $E_{\mathbf{t}|\mathbf{s}}$, it would not give us the nested expectation $E_{\mathbf{t}|\mathbf{s}}[E_{\mathbf{s}'|\mathbf{t}}]$; this is the classic challenge in training globally-normalized log-linear models without labeled data [82]. Second, both forward-backward and the Viterbi algorithm have time complexity that is quadratic in the dimension of the label space, at least 10^4 or 10^5. As we will show, Sequential Monte Carlo (SMC) algorithms have a number of advantages in this setting: they permit the efficient computation of both the outer and inner expectations, they are trivially parallelizable, and the number of samples provides an intuitive tuning tradeoff between accuracy and speed.

3.2.1 Sequential Monte Carlo approximation

Sequential Monte Carlo algorithms are a class of sampling-based algorithms in which latent variables are sampled sequentially [84]. They are particularly well-suited to sequence models, though they can be applied more broadly. SMC algorithms maintain a set of weighted hypotheses; the weights correspond to probabilities, and in our case, the hypotheses correspond to target language word sequences. Specifically, we approximate the conditional probability,

$$P(\mathbf{t}_{1:n}|\mathbf{s}_{1:n}) \approx \sum_{k=1}^{K} \omega_n^k \delta_{\mathbf{t}_{1:n}^k}(\mathbf{t}_{1:n}),$$

where ω_n^k is the normalized weight of sample k at word n ($\tilde{\omega}_n^k$ is the unnormalized weight), and $\delta_{\mathbf{t}_{1:n}^k}$ is a delta function centered at $\mathbf{t}_{1:n}^k$.

At each step, and for each hypothesis k, a new target word is sampled from a *proposal*

distribution, and the weight of the hypothesis is then updated. We maintain feature counts for each hypothesis, and approximate the expectation by taking a weighted average using the hypothesis weights. The proposal distribution will be described in detail later.

We make a Markov assumption, so that the emission probability $P(\mathbf{s}|\mathbf{t})$ decomposes across the elements of the sentence $P(\mathbf{s}|\mathbf{t}) = \prod_n^N P(s_n|t_n)$. This means that the feature functions $\mathbf{f}(\mathbf{s}, \mathbf{t})$ must decompose on each $\langle s_n, t_n \rangle$ pair. We can then rewrite (3.1) as

$$P(\mathbf{s}|\mathbf{t}; \theta) = \prod_n^N \frac{\exp\left(\theta^\top \mathbf{f}(s_n, t_n)\right)}{Z(t_n)} \tag{3.3}$$

$$Z(t_n) = \sum_s \exp\left(\theta^\top \mathbf{f}(s, t_n)\right). \tag{3.4}$$

In addition, we assume that the target language model $P(\mathbf{t})$ can be written as an N-gram language model, $P(\mathbf{t}) = \prod_n P(t_n|t_{n-1}, \ldots t_{n-k+1})$. With these assumptions, we can view normalization as a finite state-space model in which the target language model defines the prior distribution of the process and Equation 3.3 defines the likelihood function. We are able to compute the posterior probability $P(\mathbf{t}|\mathbf{s})$ using *sequential importance sampling*, a member of the SMC family.

The crucial idea in sequential importance sampling is to update the hypotheses $\mathbf{t}_{1:n}^k$ and their weights ω_n^k so that they approximate the posterior distribution at the next time step, $P(\mathbf{t}_{1:n+1}|\mathbf{s}_{1:n+1})$. Assuming the proposal distribution has the form $Q(\mathbf{t}_{1:n}^k|\mathbf{s}_{1:n})$, the importance weights are given by

$$\omega_n^k \propto \frac{P(\mathbf{t}_{1:n}^k|\mathbf{s}_{1:n})}{Q(\mathbf{t}_{1:n}^k|\mathbf{s}_{1:n})} \tag{3.5}$$

In order to update the hypotheses recursively, we rewrite $P(\mathbf{t}_{1:n}|\mathbf{s}_{1:n})$ as:

$$\begin{aligned}
P(\mathbf{t}_{1:n}|\mathbf{s}_{1:n}) &= \frac{P(s_n|\mathbf{t}_{1:n}, \mathbf{s}_{1:n-1})P(\mathbf{t}_{1:n}|\mathbf{s}_{1:n-1})}{P(s_n|\mathbf{s}_{1:n-1})} \\
&= \frac{P(s_n|t_n)P(t_n|\mathbf{t}_{1:n-1}, \mathbf{s}_{1:n-1})P(\mathbf{t}_{1:n-1}|\mathbf{s}_{1:n-1})}{P(s_n|\mathbf{s}_{1:n-1})} \\
&\propto P(s_n|t_n)P(t_n|t_{n-1})P(\mathbf{t}_{1:n-1}|\mathbf{s}_{1:n-1}),
\end{aligned}$$

assuming a bigram language model. We further assume the proposal distribution Q can be factored as:

$$\begin{aligned}
Q(\mathbf{t}_{1:n}|\mathbf{s}_{1:n}) &= Q(t_n|\mathbf{t}_{1:n-1}, \mathbf{s}_{1:n})Q(\mathbf{t}_{1:n-1}|\mathbf{s}_{1:n-1}) \\
&= Q(t_n|t_{n-1}, s_n)Q(\mathbf{t}_{1:n-1}|\mathbf{s}_{1:n-1}). \qquad (3.6)
\end{aligned}$$

Then the unnormalized importance weights simplify to a recurrence:

$$\begin{aligned}
\tilde{\omega}_n^k &= \frac{P(s_n|t_n^k)P(t_n^k|t_{n-1}^k)P(\mathbf{t}_{1:n-1}^k|\mathbf{s}_{1:n-1})}{Q(t_n|t_{n-1}, s_n)Q(\mathbf{t}_{1:n-1}^k|\mathbf{s}_{1:n-1})} \qquad (3.7) \\
&= \omega_{n-1}^k \frac{P(s_n|t_n^k)P(t_n^k|t_{n-1}^k)}{Q(t_n|t_{n-1}, s_n)} \qquad (3.8)
\end{aligned}$$

Therefore, we can approximate the posterior distribution $P(t_n|\mathbf{s}_{1:n}) \approx \sum_{k=1}^K \omega_n^k \delta_{t_n^k}(t_n)$, and compute the outer expectation as follows:

$$E_{\mathbf{t}|\mathbf{s}}[\mathbf{f}(\mathbf{s}, \mathbf{t})] = \sum_{k=1}^K \omega_N^k \sum_{n=1}^N \mathbf{f}(s_n, t_n^k) \qquad (3.9)$$

We compute the nested expectation using a non-sequential Monte Carlo approximation,

assuming we can draw $s^{\ell,k} \sim P(s|t_n^k)$.

$$E_{\mathbf{s}|\mathbf{t}^k}[\mathbf{f}(\mathbf{s}, \mathbf{t}^k)] = \frac{1}{L} \sum_{n=1}^{N} \sum_{\ell=1}^{L} \mathbf{f}(s_n^{\ell,k}, t_n^k)$$

This gives the overall gradient computation:

$$E_{\mathbf{t}|\mathbf{s}}[\mathbf{f}(\mathbf{s}, \mathbf{t}) - E_{\mathbf{s}'|\mathbf{t}}[\mathbf{f}(\mathbf{s}', \mathbf{t})]] = \frac{1}{\sum_{k=1}^{K} \tilde{\omega}_N^k} \sum_{k=1}^{K} \tilde{\omega}_N^k \times \sum_{n=1}^{N} \left(\mathbf{f}(s_n, t_n^k) - \frac{1}{L} \sum_{\ell=1}^{L} \mathbf{f}(s_n^{\ell,k}, t_n^k) \right) \tag{3.10}$$

where we sample t_n^k and update ω_n^k while moving from left-to-right, and sample $s_n^{\ell,k}$ at each n. Note that although the sequential importance sampler moves left-to-right like a filter, we use only the final weights ω_N to compute the expectation. Thus, the resulting expectation is based on the distribution $P(\mathbf{s}_{1:N}|\mathbf{t}_{1:N})$, so that no backwards "smoothing" pass [85] is needed to eliminate bias. Other applications of sequential Monte Carlo make use of resampling [84] to avoid degeneration of the hypothesis weights, but we found this to be unnecessary due to the short length of Twitter messages.

3.2.2 Proposal distribution

The major computational challenge for dynamic programming approaches to normalization is the large label space, equal to the size of the target vocabulary. It may appear that all we have gained by applying sequential Monte Carlo is to convert a computational problem into a statistical one: a naive sampling approach will have little hope of finding the small high-probability region of the high-dimensional label space. However, sequential importance sampling allows us to address this issue through the proposal distribution, from which we sample the candidate words t_n. Careful design of the proposal distribution can guide sampling towards the high-probability space. In the asymptotic limit of an infinite number of samples, any non-pathological proposal distribution will ultimately arrive at the desired

estimate, but a good proposal distribution can greatly reduce the number of samples needed.

Doucet *et al.* [86] note that the optimal proposal—which minimizes the variance of the importance weights conditional on $\mathbf{t}_{1:n-1}$ and $\mathbf{s}_{1:n}$—has the following form:

$$Q(t_n^k | s_n, t_{n-1}^k) = \frac{P(s_n|t_n^k)P(t_n^k|t_{n-1}^k)}{\sum_{t'} P(s_n|t')P(t'|t_{n-1}^k)} \tag{3.11}$$

Sampling from this proposal requires computing the normalized distribution $P(s_n|t_n^k)$; similarly, the update of the hypothesis weights (Equation 3.8) requires the calculation of Q in its normalized form. In each case, the total cost is the product of the vocabulary sizes, $\mathcal{O}(\#|\nu_T|\#|\nu_S|)$, which is not tractable as the vocabularies become large.

In low-dimensional settings, a convenient solution is to set the proposal distribution equal to the transition distribution, $Q(t_n^k|s_n, t_{n-1}^k) = P(t_n^k|t_{n-1}^k, \ldots, t_{n-k+1}^k)$. This choice is called the "bootstrap filter," and it has the advantage that the weights $\omega^{(k)}$ are exactly identical to the product of emission likelihoods $\prod_n P(s_n|t_n^k)$. The complexity of computing the hypothesis weights is thus $\mathcal{O}(\#|\nu_S|)$. However, because this proposal ignores the emission likelihood, the bootstrap filter has very little hope of finding a high-probability sample in high-entropy contexts.

We strike a middle ground between efficiency and accuracy, using a proposal distribution that is closely related to the overall likelihood, yet is tractable to sample and compute:

$$\begin{aligned} Q(t_n^k | s_n, t_{n-1}^k) &\stackrel{def}{=} \frac{P(s_n|t_n^k)Z(t_n^k)P(t_n^k|t_{n-1}^k)}{\sum_{t'} P(s_n|t')Z(t')P(t'|t_{n-1}^k)} \\ &= \frac{\exp\left(\theta^\top \mathbf{f}(s_n, t_n)\right) P(t_n^k|t_{n-1}^k)}{\sum_{t'} \exp\left(\theta^\top \mathbf{f}(s_n, t')\right) P(t'|t_{n-1}^k)} \end{aligned} \tag{3.12}$$

Here, we simply replace the likelihood distribution in (3.11) by its unnormalized version.

To update the unnormalized hypothesis weights $\tilde{\omega}_n^k$, we have

$$\tilde{\omega}_n^k = \omega_{n-1}^k \frac{\sum_{t'} \exp\left(\theta^\top \mathbf{f}(s_n, t')\right) P(t'|t_{n-1}^k)}{Z(t_n^k)} \tag{3.13}$$

The numerator requires summing over all elements in ν_T and the denominator $Z(t_n^k)$ requires summing over all elements in ν_S, for a total cost of $\mathcal{O}(\#|\nu_T| + \#|\nu_S|)$.

3.2.3 Decoding

Given an input source sentence $\mathbf{s}$, the decoding problem is to find a target sentence $\mathbf{t}$ that maximizes $P(\mathbf{t}|\mathbf{s}) \propto P(\mathbf{s}|\mathbf{t})P(\mathbf{t}) = \prod_n^N P(s_n|t_n)P(t_n|t_{n-1})$. As with learning, we cannot apply the usual dynamic programming algorithm (Viterbi), because of its quadratic cost in the size of the target language vocabulary. This must be multiplied by the cost of computing the normalized probability $P(s_n|t_n)$, resulting in a prohibitive time complexity of $\mathcal{O}(\#|\nu_S|\#|\nu_T|^2 N)$.

We consider two approximate decoding algorithms. The first is to simply apply the proposal distribution, with linear complexity in the size of the two vocabularies. However, this decoder is not identical to $P(\mathbf{t}|\mathbf{s})$, because of the extra factor of $Z(t)$ in the numerator. Alternatively, we can apply the proposal distribution for selecting target word candidates, then apply the Viterbi algorithm only within these candidates. The total cost is $\mathcal{O}(\#|\nu_S|T^2 N)$, where T is the number of target word candidates we consider; this will asymptotically approach $P(\mathbf{t}|\mathbf{s})$ as $T \to \#|\nu_T|$. Our evaluations use the more expensive proposal+Viterbi decoding, but accuracy with the more efficient proposal-based decoding is very similar.

3.2.4 Features

Our system uses the feature types described in Table 3.1. The word pair features are designed to capture lexical conventions, e.g. *you*/`u`. We only consider word pair features that fired during training. The string similarity features rely on the similarity function proposed

Table 3.1: The feature set for our log-linear model

Feature name	Description
word-word pair	A set of binary features for each source/target word pair $\langle s, t \rangle$
string similarity	A set of binary features indicating whether s is one of the top N string similar nonstandard words of t, for $N \in \{5, 10, 25, 50, 100, 250, 500, 1000\}$

by Contractor *et al.* [69], which has proven effective for normalization in prior work. We bin this similarity to create binary features indicating whether a string s is in the top-N most similar strings to t; this binning yields substantial speed improvements without negatively impacting accuracy.

3.3 Implementation and Data

The model and inference described in the previous section are implemented in a software system for normalizing text on twitter, called UNLOL: **u**nsupervised **n**ormalization in a **LO**g-**L**inear model. The final system can process roughly 10,000 Tweets per hour. We now describe some implementation details.

3.3.1 Normalization candidates

Most tokens in tweets do not require normalization. The question of how to identify which words are to be normalized is still an open problem. Following Han *et al.* [24], we build a dictionary of words which are permissible in the target domain, and make no attempliu2012broadt to normalize source strings that match these words. As with other comparable approaches, we are therefore unable to normalize strings like `ill` into *I'll*. Our set of "in-vocabulary" (IV) words is based on the GNU aspell dictionary (v0.60.6), containing 97,070 words. From this dictionary, we follow Liu *et al.* [72] and remove all the words with a count of less than 20 in the Edinburgh Twitter corpus [87]—resulting in a total of 52,449 target words. All single characters except `a` and `i` are excluded, and `rt` is treated as in-vocabulary. For all in-vocabulary words, we define $P(s_n|t_n) = \delta(s_n, t_n)$, taking the value

of zero when $s_n \neq t_n$. This effectively prevents our model from attempting to normalize these words.

In addition to words that are in the target vocabulary, there are many other strings that should not be normalized, such as names and multiword shortenings (e.g. *going to*/`gonna`).[1] We follow prior work and assume that the set of normalization candidates is known in advance during test set decoding [19]. However, the unlabeled training data has no such information. Thus, during training we attempt to normalize all tokens that (1) are not in our lexicon of IV words, and (2) are composed of letters, numbers and the apostrophe. This set includes contractions like "gonna" and "gotta", which would not appear in the test set, but are nonetheless normalized during training. For each OOV token, we conduct a pre-normalization step by reducing any repetitions of more than two letters in the nonstandard words to exactly two letters (e.g., `cooool` $\rightarrow$ `cool`).

3.3.2 Language modeling

The Kneser-Ney smoothed trigram target language model is estimated with the SRILM toolkit Stolcke [88], using Tweets from the Edinburgh Twitter corpus that contain no OOV words besides hashtags and username mentions (following [19]). We use this language model for both training and decoding. We occasionally find training contexts in which the trigram $\langle t_n, t_{n-1}, t_{n-2} \rangle$ is unobserved in the language model data; features resulting from such trigrams are not considered when computing the weight gradients.

3.3.3 Parameters

The Monte Carlo approximations require two parameters: the number of samples for sequential Monte Carlo (K), and the number of samples for the non-sequential sampler of the nested expectation (L, from Equation 3.10). The theory of Monte Carlo approximation states that the quality of the approximation should only improve as the number of samples

[1] Whether multiword shortenings should be normalized is arguable, but they are outside the scope of current normalization datasets [24].

Table 3.2: Empirical results

Method	Dataset	Precision	Recall	F-measure
(Liu *et al.* 2011)	LMML11	68.88	68.88	68.88
(Liu *et al.* 2012)		69.81	69.81	69.81
UNLOL		**73.04**	**73.04**	**73.04**
(Han and Baldwin, 2011)	LexNorm 1.1	75.30	75.30	75.30
(Liu *et al.* 2012)		84.13	78.38	81.15
(Hassan *et al.* 2013)		**85.37**	56.4	69.93
UNLOL		82.09	**82.09**	**82.09**
UNLOL	LexNorm 1.2	82.06	82.06	82.06

increases; we obtained good results with $K = 10$ and $L = 1$, and found relatively little improvement by increasing these values. The number of hypotheses considered by the decoder is set to $T = 10$; again, the performance should only improve with T, as we more closely approximate full Viterbi decoding.

3.4 Experiments

Datasets We use two existing labeled Twitter datasets to evaluate our approach. The first dataset—which we call LWWL11, based on the names of its authors Liu *et al.* [26]—contains 3,802 individual "nonstandard" words (i.e., words that are not in the target vocabulary) and their normalized forms. The rest of the message in which the words is appear is not available. As this corpus does not provide linguistic context, its decoding must use a unigram target language model. The second dataset—which is called LexNorm1.1 by its authors Han *et al.* [24]—contains 549 complete tweets with 1,184 nonstandard tokens (558 unique word types). In this corpus, we can decode with a trigram language model.

Close analysis of LexNorm1.1 revealed some inconsistencies in annotation (for example, `y'all` and `2` are sometimes normalized to *you* and *to*, but are left unnormalized in other cases). In addition, several annotations disagree with existing resources on internet language and dialectal English. For example, `smh` is normalized to *somehow* in LexNorm1.1, but `internetslang.com` and `urbandictionary.com` assert that it

stands for *shake my head*, and this is evident from examples such as `smh at this girl`. Similarly, `finna` is normalized to *finally* in LexNorm1.1, but from the literature on African American English [51], it corresponds to *fixing to* (e.g., `i'm finna go home`). To address these issues, we have produced a new version of this dataset, which we call LexNorm1.2 (after consulting with the creators of LexNorm1.1). LexNorm1.2 differs from version 1.1 in the annotations for 172 of the 2140 OOV words. We evaluate on LexNorm1.1 to compare with prior work, but we also present results on LexNorm1.2 in the hope that it will become standard in future work on normalization in English. The dataset is available at `http://www.cc.gatech.edu/~jeisenst/lexnorm.v1.2.tgz`.

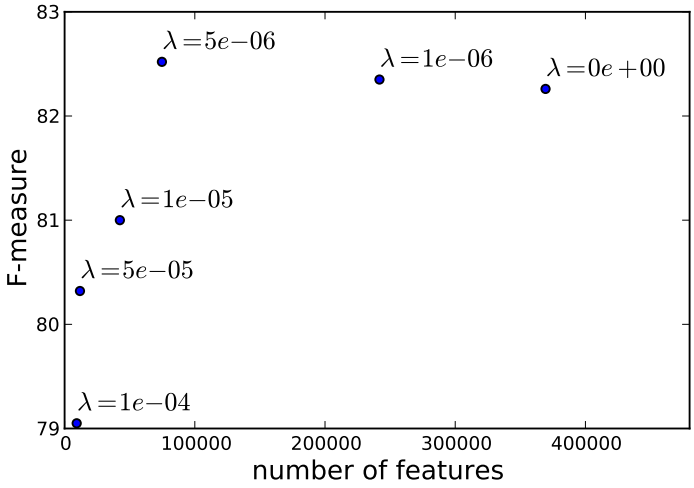
To obtain unlabeled training data, we randomly sample 50 tweets from the Edinburgh Twitter corpus Petrović *et al.* [87] for each OOV word. Some OOV words appear less than 50 times in the corpus, so we obtained more training tweets for them through the Twitter search API.

Metrics Prior work on these datasets has assumed perfect detection of words requiring normalization, and has focused on finding the correct normalization for these words [24, 19]. Recall has been defined as the proportion of words requiring normalization which are normalized correctly; precision is defined as the proportion of normalizations which are correct.

Results We run our training algorithm for two iterations (pass the training data twice). The results are presented in Table 3.2. Our system, UNLOL, achieves the highest published F-measure on both datasets. Performance on LexNorm1.2 is very similar to LexNorm1.1, despite the fact that roughly 8% of the examples were relabeled.

In the normalization task that we consider, the tokens to be normalized are specified in advance. This is the same task specification as in the prior work against which we compare. At test time, our system attempts normalizes all such tokens; every error is thus both a false positive and false negative, so precision equals to recall for this task; this is also true for

Table 3.3: Effect of L1 regularization on the F-measure and the number of non-zero features



λ	Dataset	F-measure	# of features
10^{-4}	LexNorm 1.1	79.05	9,281
5×10^{-5}		80.32	11,794
10^{-5}		81.00	42,466
5×10^{-6}		**82.52**	74,744
10^{-6}		82.35	241,820
0		82.26	369,366
5×10^{-6}	LexNorm 1.2	82.23	74,607

Han *et al.* [24] and Liu *et al.* [26].

It is possible to trade recall for precision by refusing to normalize words when the system‘s confidence falls below a threshold. A good setting of this threshold can improve the F-measure, but we did not report these results because we have no development set for parameter tuning.

Regularization One potential concern is that the number of non-zero feature weights will continually increase until the memory cost becomes overwhelming. Although we did not run up against memory limitations in the experiments producing the results in Table 3.2, this issue can be addressed through the application of L1 regularization, which produces sparse weight vectors by adding a penalty of $\lambda||\theta||_1$ to the log-likelihood. We perform

online optimization of the L1-regularized log-likelihood by applying the truncated gradient method [89]. We use an exponential decreasing learning rate $\eta_k = \eta_0 \alpha^{k/N}$, where k is the iteration counter and N is the size of training data. We set $\eta_0 = 1$ and $\alpha = 0.5$. Experiments were run until 300,000 training instances were observed, with a final learning rate of less than $1/32$. As shown in Figure 3.3, a small amount of regularization can dramatically decrease the number of active features without harming performance.

3.5 Analysis

We apply our normalization system to investigate the orthographic processes underlying language variation in social media. Using a dataset of 400,000 English language tweets, sampled from the month of August in each year from 2009 to 2012, we apply UNLOL to automatically normalize each token. We then treat these normalizations as labeled training data, and examine the Levenshtein alignment between the source and target tokens. This alignment gives approximate character-level transduction rules to explain each OOV token. We then examine which rules are used by each author, constructing a matrix of authors and rules.[2]

Factorization of the author-rule matrix reveals sets of rules that tend to be used together; we might call these rulesets "orthographic styles." We apply non-negative matrix factorization [90], which characterizes each author by a vector of k style loadings, and simultaneously constructs k style dictionaries, which each put weight on different orthographic rules. Because the loadings are constrained to be non-negative, the factorization can be seen as sparsely assigning varying amounts of each style to each author. We choose the factorization that minimizes the Frobenius norm of the reconstruction error, using the NIMFA software package (`http://nimfa.biolab.si/`).

The resulting styles are shown in Table 3.4, for $k = 10$; other values of k give similar overall results with more or less detail. The styles incorporate a number of linguistic phe-

[2] We tried adding these rules as features and retraining the normalization system, but this hurt performance.

Table 3.4: Orthographic styles induced from automatically normalized Twitter text

style	rules	examples
1. you; o-dropping	*y*/_ *ou*/_u **y*/⋆_ *o*/_	u, yu, 2day, knw, gud, yur, wud, yuh, u've, toda, everthing, everwhere, ourself
2. e-dropping, u/o	*be*/b_ *e*/_ *o*/u *e**/_⋆	b, r, luv, cum, hav, mayb, bn, remembr, btween, gunna, gud
3. a-dropping	*a*/_ **a*/⋆_ *re*/r_ *ar*/_r	r, tht, wht, yrs, bck, strt, gurantee, elementry, wr, rlly, wher, rdy, preciate, neway
4. g-dropping	*g**/_⋆ *ng*/n_ *g*/_	goin, talkin, watchin, feelin, makin
5. t-dropping	*t**/_⋆ *st*/s_ *t*/_	jus, bc, shh, wha, gota, wea, mus, firts, jes, subsistutes
6. th-stopping	*h*/_ **t*/⋆d *th*/d_ *t*/d	dat, de, skool, fone, dese, dha, shid, dhat, dat's
7. (kd)-lengthening	*i*_/id _/k _/d _*/k⋆	idk, fuckk, okk, backk, workk, badd, andd, goodd, bedd, elidgible, pidgeon
8. o-lengthening	*o*_/oo _*/o⋆ _/o	soo, noo, doo, oohh, loove, thoo, helloo
9. e-lengthening	_/i *e*_/ee _/e _*/e⋆	mee, ive, retweet, bestie, lovee, nicee, heey, likee, iphone, homie, ii, damnit
10. a-adding	_/a __/ma _/m _*/a⋆	ima, outta, needa, shoulda, woulda, mm, comming, tomm, boutt, ppreciate

nomena, including: expressive lengthening (styles 7-9; see Brody and Diakopoulos [91]); g- and t-dropping (style 5, see Eisenstein [92]); th-stopping (style 6); and the dropping of several word-final vowels (styles 1-3). Some of these styles, such as t-dropping and th-stopping, have direct analogues in spoken language varieties [93, 51], while others, like expressive lengthening, seem more unique to social media. The relationships between these orthographic styles and social variables such as geography and demographics must be left to future research, but they offer a promising generalization of prior work that has focused almost exclusively on exclusively on lexical variation [94, 2, 95], with a few exceptions for character-level features [91, 96].

Note that style 10 is largely the result of mistaken normalizations. The tokens `ima`, `outta`, and `needa` all refer to multi-word expressions in standard English, and are thus outside the scope of the normalization task as defined by Han *et al.* [19]. UNLOL has produced incorrect single-token normalizations for these terms: *i*/`ima`, *out*/`outta`, and *need*/`needa`. But while these normalizations are wrong, the resulting style nonetheless capture a coherent orthographic phenomenon.

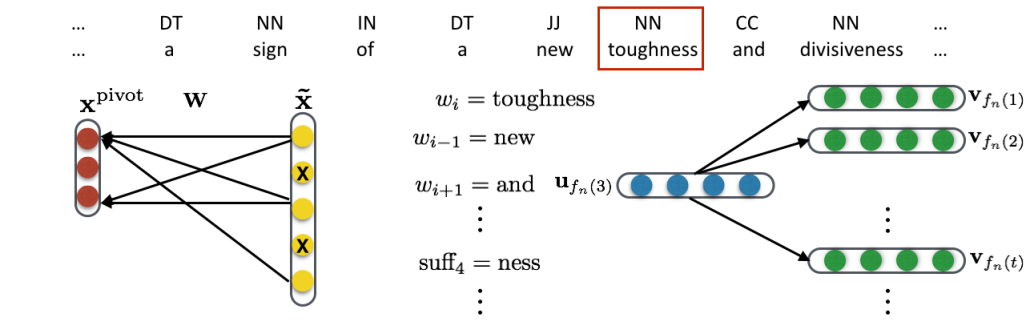
CHAPTER 4

DOMAIN ADAPTATION WITH METADATA ATTRIBUTES

In this chapter we present a novel but simple approach for unsupervised domain adaptation, called Feature Embedding. In contrast to text normalization, domain adaptation intends to overcome linguistic variation with respect to specific NLP tasks. Cross-domain feature representations are induced to tighten the representational differences between source domain and target domain data for tasks like part-of-speech (POS) tagging, named entity recognition (NER), and sentiment analysis. We primally demonstrate the utility of Feature Embedding on the task of part-of-speech tagging in this chapter, but it can be easily adopted to other NLP tasks as well.

As mentioned in chapter 2, some of the most successful approaches to unsupervised domain adaptation are based on representation learning: transforming sparse high-dimensional surface features into dense vector representations, which are often more robust to domain shift [77, 97]. However, these methods are computationally expensive to train, because they need to work with the $D \times D$ feature co-occurrence matrix, where D is the number of surface features. To pursue a tradeoff between efficiency and effectiveness, these approaches often require special task-specific heuristics to select good *pivot features*.

A second, more subtle challenge for unsupervised domain adaptation is that it is normally framed as adapting from a single source domain to a single target domain. For example, as shown in Figure 1.5, we may be given part-of-speech labeled text from 19th Century narratives, and we hope to adapt the tagger to work on academic dissertations from the 16th Century. This ignores text from the intervening centuries, as well as text that is related by genre, such as 16th Century narratives and 19th Century dissertations. We address a new challenge of *unsupervised multi-domain adaptation*, where the goal is to leverage



(a) Learning a dense transformation $\mathbf{W}$ by reconstructing pivot features.

(b) Directly learning of feature embeddings by maximizing Skip-Gram objective.

Figure 4.1: Representation learning techniques in structured feature spaces

this additional unlabeled data to improve performance in the target domain.[1]

We present FEMA (**F**eature **EM**beddings for domain **A**daptation), a novel representation learning approach for domain adaptation in structured feature spaces. Like prior work in representation learning, FEMA learns dense features that are more robust to domain shift. However, rather than performing representation learning by reconstructing pivot features, FEMA uses techniques from neural language models to obtain low-dimensional embeddings directly. FEMA outperforms prior work on adapting POS tagging from the Penn Treebank to web text, and it easily generalizes to unsupervised multi-domain adaptation, further improving performance by learning generalizable models across multiple domains. In addition, we demonstrate that the Feature Embedding method for unsupervised domain adaptation outperforms spelling normalization on adapting POS tagging from the Penn Treebank to the Penn Corpora of Historical English [99, 100]. The combination of the two methods is better still, yielding a 5% raw improvement in tagging accuracy on Early Modern English texts.

4.1 Learning Feature Embeddings

Feature co-occurrence statistics are the primary source of information driving many unsupervised methods for domain adaptation; they enable the induction of representations that

[1] Multiple domains have been considered in **supervised** domain adaptation (e.g., Mansour *et al.* [98]), but these approaches are not directly applicable when there is no labeled data outside the source domain.

are more similar across the source and target domain, reducing the error introduced by domain shift [75]. For example, both Structural Correspondence Learning (SCL; Blitzer *et al.*, 2006) and Denoising Autoencoders [78] learn to reconstruct a subset of *pivot features*, as shown in Figure 4.1(a). The reconstruction function—which is learned from unlabeled data in both domains—is then employed to project each instance into a dense representation, which will hopefully be better suited to cross-domain generalization. The pivot features are chosen to be both predictive of the label and general across domains. Meeting these two criteria requires task-specific heuristics; for example, different pivot selection techniques are employed in SCL for syntactic tagging [77] and sentiment analysis [101]. Furthermore, the pivot features correspond to a small subspace of the feature co-occurrence matrix. In Denoising Autoencoders, each pivot feature corresponds to a dense feature in the transformed representation, but large dense feature vectors impose substantial computational costs at learning time. In SCL, each pivot feature introduces a new classification problem, which makes computation of the cross-domain representation expensive. In either case, we face a tradeoff between the amount of feature co-occurrence information that we can use, and the computational complexity for representation learning and downstream training.

This tradeoff can be avoided by inducing low dimensional feature embeddings directly. We exploit the tendency of many NLP tasks to divide features into *templates*, with exactly one active feature per template [102]; this is shown in the center of Figure 4.1. Rather than treating each instance as an undifferentiated bag-of-features, we use this template structure to induce *feature embeddings*, which are dense representations of individual features. Each embedding is selected to help predict the features that fill out the other templates: for example, an embedding for the current word feature is selected to help predict the previous word feature and successor word feature, and vice versa; see Figure 4.1(b). The embeddings for each active feature are then concatenated together across templates, giving a dense representation for the entire instance.

Our approach is motivated by word embeddings, in which dense representations are learned for individual words based on their neighbors [103, 104], but rather than learning a single embedding for each word, we learn embeddings for each *feature*. This means that the embedding of, say, 'toughness' will differ depending on whether it appears in the current-word template or the previous-word template (see Table 4.13). This provides additional flexibility for the downstream learning algorithm, and the increase in the dimensionality of the overall dense representation can be offset by learning shorter embeddings for each feature. In section 4.3, we show that feature embeddings convincingly outperform word embeddings on two part-of-speech tagging tasks.

Our feature embeddings are based on the skip-gram model, trained with negative sampling [105], which is a simple yet efficient method for learning word embeddings. Rather than predicting adjacent words, the training objective in our case is to find feature embeddings that are useful for predicting other active features in the instance. For the instance $n \in \{1 \dots N\}$ and feature template $t \in \{1 \dots T\}$, we denote $f_n(t)$ as the index of the active feature; for example, in the instance shown in Figure 4.1, $f_n(t) =$ 'new' when t indicates the previous-word template. The skip-gram approach induces distinct "input" and "output" embeddings for each feature, written $\mathbf{u}_{f_n(t)}$ and $\mathbf{v}_{f_n(t)}$, respectively. The role of these embeddings can be seen in the negative sampling objective,

$$\ell_n = \frac{1}{T} \sum_{t=1}^{T} \sum_{t' \neq t}^{T} \left[\log \sigma(\mathbf{u}_{f_n(t)}^\top \mathbf{v}_{f_n(t')}) + k \mathbb{E}_{i \sim P_{t'}^{(n)}} \log \sigma(-\mathbf{u}_{f_n(t)}^\top \mathbf{v}_i) \right], \tag{4.1}$$

where t and t' are feature templates, k is the number of negative samples, $P_{t'}^{(n)}$ is a *noise distribution* for template t', and σ is the sigmoid function. This objective is derived from noise-contrastive estimation [106], and is chosen to maximize the unnormalized log-likelihood of the observed feature co-occurrence pairs, while minimizing the unnormalized log-likelihood of "negative" samples, drawn from the noise distribution.

Feature embeddings can be applied to domain adaptation by learning embeddings of

all features on the union of the source and target data sets; we consider the extension to multiple domains in the next section. The dense feature vector for each instance is obtained by concatenating the feature embeddings for each template. Finally, since it has been shown that nonlinearity is important for generating robust representations [107], we follow Chen *et al.* [78] and apply the hyperbolic tangent function to the embeddings. The augmented representation $\mathbf{x}_n^{(\text{aug})}$ of instance n is the concatenation of the original feature vector and the feature embeddings,

$$\mathbf{x}_n^{(\text{aug})} = \mathbf{x}_n \oplus \tanh[\mathbf{u}_{f_n(1)} \oplus \cdots \oplus \mathbf{u}_{f_n(T)}],$$

where $\oplus$ is vector concatenation.

4.2 Feature Embeddings Across Domains

We now describe how to extend the feature embedding idea beyond a single source and target domain, to unsupervised multi-attribute domain adaptation [108]. In this setting, each instance is associated with M metadata domain attributes, which could encode temporal epoch, genre, or other aspects of the domain. The challenge of domain adaptation is that the meaning of features can shift across each metadata dimension: for example, the meaning of 'plant' may depend on genre (agriculture versus industry), while the meaning of 'like' may depend on epoch. To account for this, the feature embeddings should smoothly shift over domain graphs, such as the one shown in Figure 1.5; this would allow us to isolate the domain general aspects of each feature. Related settings have been considered only for supervised domain adaptation, where some labeled data is available in each domain [108], but not in the unsupervised case.

More formally, we assume each instance n is augmented with a vector of M binary domain attributes, $\mathbf{z}_n \in \{0, 1\}^M$. These attributes may overlap, so that we could have an attribute for the epoch 1800-1849, and another for the epoch 1800-1899. We define

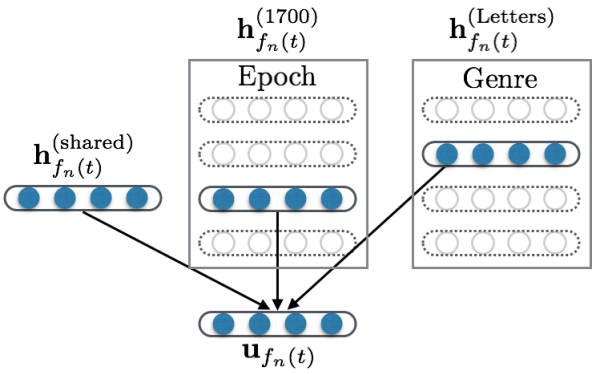


Figure 4.2: Aggregating multiple embeddings.

$z_{n,0} = 1$ as a shared attribute, which is active for all instances. We capture domain shift by estimating embeddings $\mathbf{h}_i^{(m)}$ for each feature i crossed with each domain attribute m. We then compute the embedding for each instance by summing across the relevant domain attributes, as shown in Figure 4.2. The local "input" feature embedding $\mathbf{u}_{f_n(t)}$ is then defined as the summation, $\mathbf{u}_{f_n(t)} = \sum_{m=0}^{M} z_{n,m}\mathbf{h}^{(m)}_{f_n(t)}$.

The role of the global embedding $\mathbf{h}_i^{(0)}$ is to capture domain-neutral information about the feature i, while the other embeddings capture attribute-specific information. The global feature embeddings should therefore be more robust to domain shift, which is "explained away" by the attribute-specific embeddings. We therefore use only these embeddings when constructing the augmented representation, $\mathbf{x}_n^{(\text{aug})}$. To ensure that the global embeddings capture all of the domain-general information about each feature, we place an L2 regularizer on the attribute-specific embeddings. Note that we do not learn attribute-specific "output" embeddings $\mathbf{v}$; these are shared across all instances, regardless of domain.

The attribute-based embeddings yield a new training objective for instance n,

$$\ell_n = \frac{1}{T}\sum_{t=1}^{T}\sum_{t'\neq t}^{T}\Bigg[\log\sigma([\sum_{m=0}^{M} z_{n,m}\mathbf{h}^{(m)}_{f_n(t)}]^\top \mathbf{v}_{f_n(t')}) + k\mathbb{E}_{i\sim P_{t'}^{(n)}}\log\sigma(-[\sum_{m=0}^{M} z_{n,m}\mathbf{h}^{(m)}_{f_n(t)}]^\top \mathbf{v}_i)\Bigg]. \tag{4.2}$$

For brevity, we omit the regularizer from Equation 4.2. For feature $f_n(t)$, the (unregular-

ized) gradients of $\mathbf{h}^{(m)}_{f_n(t)}$ and $\mathbf{v}_{f_n(t')}$ *w.r.t* $\ell_{n,t}$ are

$$\begin{aligned}\frac{\partial \ell_{n,t}}{\mathbf{h}^{(m)}_{f_n(t)}} =& \frac{1}{T}\sum_{t' \neq t}^{T} z_{n,m}\left[(1-\sigma(\mathbf{u}_{f_n(t)}^{\top}\mathbf{v}_{f_n(t')}))\mathbf{v}_{f_n(t')}\right. \\ & \left. -k\mathbb{E}_{i\sim P^{(n)}_{t'}}\sigma(\mathbf{u}_{f_n(t)}^{\top}\mathbf{v}_i)\mathbf{v}_i\right] \qquad (4.3)\\ \frac{\partial \ell_{n,t}}{\mathbf{v}_{f_n(t')}} =& \frac{1}{T}\sum_{t' \neq t}^{T}(1-\sigma(\mathbf{u}_{f_n(t)}^{\top}\mathbf{v}_{f_n(t')}))\mathbf{u}_{f_n(t)}. \qquad (4.4)\end{aligned}$$

For each feature i drawn from the noise distribution $P^{(n)}_{t'}$, the gradient of $\mathbf{v}_i$ *w.r.t* $\ell_{n,t}$ is

$$\frac{\partial \ell_{n,t}}{\mathbf{v}_i} = -\frac{1}{T}\sigma(\mathbf{u}_{f_n(t)}^{\top}\mathbf{v}_i)\mathbf{u}_{f_n(t)}. \qquad (4.5)$$

4.3 Experiments

We evaluate FEMA on part-of-speech (POS) tagging, in three settings: (1) adaptation of English POS tagging from news text to web text, as in the SANCL shared task [109]; (2) adaptation of Portuguese POS tagging across a graph of related domains over several centuries and genres, from the Tycho Brahe corpus [29]; (3) adaptation of English POS tagging from the Penn Treebank to the Penn Corpora of Historical English. These evaluations are complementary: English POS tagging on web text gives us the opportunity to evaluate feature embeddings in a high-impact application; Portuguese POS tagging enables evaluation of multi-attribute domain adaptation, and demonstrates the capability of our approach in a morphologically-rich language, with a correspondingly large number of part-of-speech tags (383); the last setting is the standard and well-studied evaluation scenario for POS tagging, where we train on the Wall Street Journal (WSJ) text from the Penn Treebank and test on historical English texts.

Table 4.1: Basic feature templates for token w_i.

Component	Feature template
Lexical (5)	$w_{i-2} = X, w_{i-1} = Y, \ldots$
Affixes (8)	X is prefix of w_i, $\lvert X\rvert \leq 4$; X is suffix of w_i, $\lvert X\rvert \leq 4$
Orthography (3)	w_i contains number, uppercase character, or hyphen

4.3.1 Implementation details

While POS tagging is classically treated as a structured prediction problem, we follow Schnabel *et al.* [110] by taking a classification-based approach. Feature embeddings can easily be used in feature-rich sequence labeling algorithms such as conditional random fields or structured perceptron, but our pilot experiments suggest that with sufficiently rich features, classification-based methods can be extremely competitive on these datasets, at a fraction of the computational cost. Specifically, we apply a support vector machine (SVM) classifier, adding dense features from FEMA (and the alternative representation learning techniques) to a set of basic features.

Basic features We apply sixteen feature templates, motivated by by Ratnaparkhi [111]. Table 4.1 provides a summary of the templates; there are four templates each for the prefix and suffix features. Feature embeddings are learned for all lexical and affix features, yielding a total of thirteen embeddings per instance. We do not learn embeddings for the binary orthographic features. Santos *et al.* [112] demonstrate the utility of embeddings for affix features.

Competitive systems We consider three competitive unsupervised domain adaptation methods. Structural Correspondence Learning [77, SCL] creates a binary classification problem for each pivot feature, and uses the weights of the resulting classifiers to project the instances into a dense representation. Marginalized Denoising Autoencoders [78, mDA] learn robust representation across domains by reconstructing pivot features from artificially

Table 4.2: Accuracy results for adaptation from WSJ to Web Text on SANCL dev set.

Target	baseline	MEMM	SCL	mDA	word2vec	FLORS	FEMA
NEWSGROUPS	88.56	89.11	89.33	89.87	89.70	90.86	**91.26**
REVIEWS	91.02	91.43	91.53	91.96	91.70	**92.95**	92.82
WEBLOGS	93.67	94.15	94.28	94.18	94.17	94.71	**94.95**
ANSWERS	89.05	88.92	89.56	90.06	89.83	90.30	**90.69**
EMAILS	88.12	88.68	88.42	88.71	88.51	89.44	**89.72**
AVERAGE	90.08	90.46	90.63	90.95	90.78	91.65	**91.89**

corrupted input instances. We use structured dropout noise, which has achieved state-of-art results on domain adaptation for part-of-speech tagging [113]. We also directly compare with WORD2VEC[2] word embeddings, and with a "no-adaptation" baseline in which only surface features are used.

Parameter tuning All the hyperparameters are tuned on development data. Following Blitzer *et al.* [77], we consider pivot features that appear more than 50 times in all the domains for SCL and mDA. In SCL, the parameter K selects the number of singular vectors of the projection matrix to consider; we try values between 10 and 100, and also employ feature normalization and rescaling. For embedding-based methods, we choose embedding sizes and numbers of negative samples from $\{25, 50, 100, 150, 200\}$ and $\{5, 10, 15, 20\}$ respectively. The noise distribution $P_t^{(n)}$ is simply the unigram probability of each feature in the template t. Mikolov *et al.* [114] argue for exponentiating the unigram distribution, but we find it makes little difference here. The window size of word embeddings is set as 5. As noted above, the attribute-specific embeddings are regularized, to encourage use of the shared embedding $\mathbf{h}^{(0)}$. The regularization penalty is selected by grid search over $\{0.001, 0.01, 0.1, 1.0, 10.0\}$. In general, we find that the hyperparameters that yield good word embeddings tend to yield good feature embeddings too.

[2]https://code.google.com/p/word2vec/

Table 4.3: Accuracy results for adaptation from WSJ to Web Text on SANCL test set.

Target	baseline	MEMM	SCL	mDA	word2vec	FLORS	FEMA
NEWSGROUPS	91.02	91.25	91.51	91.83	91.35	92.41	**92.60**
REVIEWS	89.79	90.30	90.29	90.95	90.87	**92.25**	92.15
WEBLOGS	91.85	92.32	92.32	92.39	92.42	93.14	**93.43**
ANSWERS	89.52	89.74	90.04	90.61	90.48	91.17	**91.35**
EMAILS	87.45	87.77	88.04	88.11	88.28	88.67	**89.02**
AVERAGE	89.93	90.28	90.44	90.78	90.68	91.53	**91.71**

4.3.2 Evaluation 1: Web text

Recent work in domain adaptation for natural language processing has focused on the data from the shared task on Syntactic Analysis of Non-Canonical Language (SANCL; Petrov and McDonald, 2012), which contains several web-related corpora (newsgroups, reviews, weblogs, answers, emails) as well as the WSJ portion of OntoNotes corpus [115]. Following Schnabel *et al.* [110], we use sections 02-21 of WSJ for training and section 22 for development, and use 100,000 unlabeled WSJ sentences from 1988 for learning representations. On the web text side, each of the five target domains has an unlabeled training set of 100,000 sentences (except the ANSWERS domain, which has 27,274 unlabeled sentences), along with development and test sets of about 1000 labeled sentences each. In the spirit of truly unsupervised domain adaptation, we do not use any target domain data for parameter tuning.

Settings For FEMA, we consider only the single-embedding setting, learning a single feature embedding jointly across all domains. We select 6918 pivot features for SCL, according to the method described above; the final dense representation is produced by performing a truncated singular value decomposition on the projection matrix that arises from the weights of the pivot feature predictors. The mDA method does not include any such matrix factorization step, and therefore generates a number of dense features equal to the number of pivot features. Memory constraints force us to choose fewer pivots, which

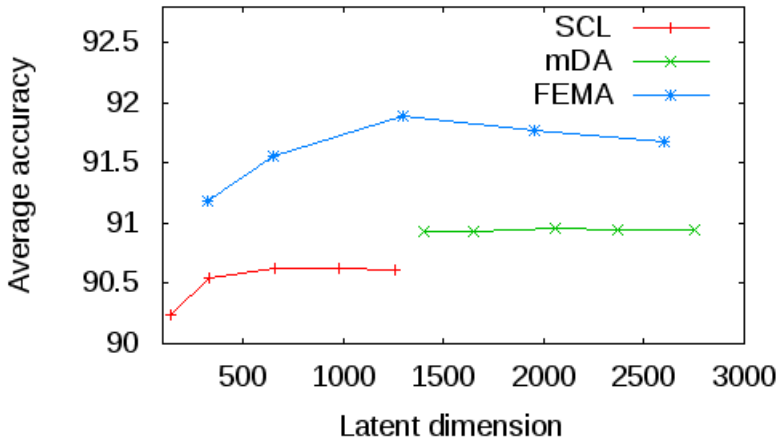


Figure 4.3: Accuracy results with different latent dimensions on SANCL dev sets.

we achieve by raising the threshold to 200, yielding 2754 pivot features.

Additional systems Aside from SCL and mDA, we compare against published results of FLORS [110], which uses distributional features for domain adaptation. We also republish the baseline results of Schnabel *et al.*[110] using the Stanford POS Tagger, a maximum entropy Markov model (MEMM) tagger.

Results As shown in Table 4.2 and 4.3, FEMA outperforms competitive systems on all target domains except REVIEW, where FLORS performs slightly better. FLORS uses more basic features than FEMA; these features could in principle be combined with feature embeddings for better performance. Compared with the other representation learning approaches, FEMA is roughly 1% better on average, corresponding to an error reduction of 10%. Its training time is approximately 70 minutes on a 24-core machine, using an implementation based on gensim.[3] This is slightly faster than SCL, although slower than mDA with structured dropout noise.

Figure 4.3 shows the average accuracy on the SANCL development set, versus the latent dimensions of different methods. The latent dimension of SCL is modulated by the number of singular vectors; we consider sizes 10, 25, 50, 75, and 100. In mDA, we

[3] `http://radimrehurek.com/gensim/`

Table 4.4: Accuracy results for adaptation in the Tycho Brahe corpus of historical Portuguese.

Task	baseline	SCL	mDA	word2vec	FEMA	
					single embedding	attribute embeddings
from 1800-1849						
→ 1750	88.74	89.31	90.11	89.24	90.25	**90.59**
→ 1700	89.97	90.41	91.39	90.51	91.61	**92.03**
→ 1650	85.94	86.76	87.69	86.22	87.64	**88.12**
→ 1600	86.21	87.65	88.63	87.41	89.39	**89.77**
→ 1550	88.92	89.92	90.79	89.85	91.47	**91.78**
→ 1500	85.32	86.82	87.64	86.60	89.29	**89.89**
AVERAGE	87.52	88.48	89.37	88.30	89.94	**90.36**
from 1750-1849						
→ 1700	94.37	94.60	94.86	94.60	95.14	**95.22**
→ 1650	91.49	91.78	92.52	91.85	92.56	**93.26**
→ 1600	91.92	92.51	93.14	92.83	93.80	**93.89**
→ 1550	92.75	93.21	93.53	93.21	**94.23**	94.20
→ 1500	89.87	90.53	91.31	91.48	92.05	**92.95**
AVERAGE	92.08	92.53	93.07	92.80	93.56	**93.90**

consider pivot feature frequency thresholds 500, 400, 300, 250, and 200. For FEMA, we consider embedding sizes 25, 50, 100, 150, and 200. The resulting latent dimensionality multiplies these sizes by the number of non-binary templates 13. FEMA dominates the other approaches across the complete range of latent dimensionalities. The best parameters for SCL are dimensionality $K = 50$ and rescale factor $\alpha = 5$. For both FEMA and WORD2VEC, the best embedding size is 100 and the best number of negative samples is 5.

4.3.3 Evaluation 2: Historical Portuguese

Next, we consider the problem of *multi-attribute domain adaptation*, using the Tycho Brahe corpus of historical Portuguese text [29], which contains syntactic annotations of Portuguese texts in four genres over several centuries (Figure 1.5). Table 4.5 presents some statistics of the datasets. We focus on temporal adaptation: training on the most modern data in the corpus, and testing on increasingly distant historical text.

Table 4.5: Statistics of the Tycho Brahe Corpus

Dataset	# of Tokens				
	Total	Narrative	Letters	Dissertation	Theatre
1800-1849	125719	91582	34137	0	0
1750-1799	202346	57477	84465	0	60404
1700-1749	278846	0	130327	148519	0
1650-1699	248194	83938	115062	49194	0
1600-1649	295154	117515	115252	62387	0
1550-1599	148061	148061	0	0	0
1500-1549	182208	126516	0	55692	0
Overall	1480528	625089	479243	315792	60404

Settings For FEMA, we consider domain attributes for 50-year temporal epochs and genres; we also create an additional attribute merging all instances that are in neither the source nor target domain. In SCL and mDA, 1823 pivot features pass the threshold. Optimizing on a source-domain development set, we find that the best parameters for SCL are dimensionality $K = 25$ and rescale factor $\alpha = 5$. The best embedding size and negative sample number are 50 and 15 for both FEMA and WORD2VEC.

Results As shown in Table 4.4, FEMA outperforms competitive systems on all tasks. The column "single embedding" reports results with a single feature embedding per feature, ignoring domain attributes; the column "attribute embeddings" shows that learning feature embeddings for domain attributes further improves performance, by 0.3-0.4% on average.

4.3.4 Evaluation 3: Historical English

Finally, we evaluate on the standard and well-studied scenario for English POS tagging. Newspaper text is the primary data source for training modern NLP systems. For example, most "off-the-shelf" English POS taggers (e.g., the Stanford Tagger [116], SVMTool [117], and CRFTagger [118]) are trained on the WSJ portion of the Penn Treebank, which is composed of professionally-written news text from 1989. This motivates this evaluation scenario, in which we train the tagger on the Penn Treebank WSJ data and apply it to

Table 4.6: Statistics of the Penn Parsed Corpus of Modern British English (PPCMBE), by time period.

Period	# Sentence	# Token
1840-1914	17,770	322,255
1770-1839	23,462	427,424
1700-1769	16,083	343,024
Total	57,315	1,092,703

historical English texts, using the Penn Corpora of Historical English.

Data

The Penn Corpora of Historical English consist of the Penn-Helsinki Parsed Corpus of Middle English, second edition [100, PPCME2], the Penn-Helsinki Parsed Corpus of Early Modern English [99, PPCEME], and the Penn Parsed Corpus of Modern British English [119, PPCMBE]. The corpora are annotated with part-of-speech tags and syntactic parsing trees in an annotation style similar to that of the Penn Treebank (PTB). In this work, we focus on POS tagging the PPCMBE and the PPCEME.[4]

The Penn Parsed Corpus of Modern British English The PPCMBE is a syntactically annotated corpus of text, containing roughly one million word tokens from documents written in the period 1700-1914. It is divided into three 70-year time periods according to the composition date of the works. Table 4.6 shows the statistics of the corpus by time period.[5] In contrast to the PTB, the PPCMBE contains text from a variety of genres, such as Bible, Drama, Fiction, and Letters.

The Penn-Helsinki Parsed Corpus of Early Modern English The PPCEME is a collection of text samples from the Helsinki Corpus [121], as well as two supplements mainly

[4]Middle English is outside the scope of this paper, because it is sufficiently unintelligible to modern English speakers that texts such as Canterbury Tales are published in translation. In tagging Middle English texts, Moon and Baldridge [120] apply bitext projection techniques from multilingual learning, rather than domain adaptation.

[5]All the statistics in this section include punctuation, but exclude extra-linguistic material such as page numbers or token ID numbers.

Table 4.7: Statistics of the Penn Parsed Corpus of Early Modern English (PPCEME), by time period.

Period	# Sentence	# Token
1640-1710	29,181	614,315
1570-1639	39,799	706,587
1500-1569	31,416	640,255
Total	100,396	1,961,157

consisting of text material by the same authors and from the same editions as the material in the Helsinki Corpus. The corpus contains nearly two million words from texts in the period from 1500 until 1710, and it is divided into three 70-year time periods similar to the PPCMBE corpus. The statistics of the corpus by time period is summarized in Table 4.7. The PPCEME consists of text from the same eighteen genres as the PPCMBE.

Penn Treebank Release 3 The Penn Treebank [57] is the de facto standard syntactically annotated corpus for English, which is used to train software such as Stanford CoreNLP [122]. When using this dataset for supervised training, we follow Toutanova *et al.* [116] and use WSJ sections 0-18 for training, and sections 19-21 for tuning. When applying unsupervised domain adaptation, we use all WSJ sections, together with texts from the PPCMBE and the PPCEME.

Tagsets The Penn Corpora of Historical English (PCHE) use a tagset that differs from the Penn Treebank, mainly in the direction of greater specificity. Auxiliary verbs 'do', 'have', and 'be' all have their own tags, as do words like 'one' and 'else', due to their changing syntactic function over time. Overall, there are 83 tags in the PPCEME, and 81 in the PPCMBE, as compared with 45 in the PTB. Furthermore, the tags in the PCHE tagset are allowed to join constituent morphemes in compounds, yielding complex tags such as `PRO+N` (e.g., 'himself') and `ADJ+NS` (e.g., 'gentlemen').

To measure the tagging accuracy of PTB-trained taggers on the historical texts, we follow Moon and Baldridge [120], who define a set of deterministic mappings from the

PCHE tags to the PTB tagset. For simplicity, we first convert each complex tag to the simple form by only considering the first simple tag component (e.g., `PRO+N` to `PRO` and `ADJ+NS` to `ADJ`). This has little effect on the tagging performance, as the complex tags cover only slightly more than 1% of the tokens in the PCHE treebanks. Among the 83 tags, 74 mappings to the corresponding PTB tags are obtained from Moon and Baldridge [120]. We did our best to convert the other tags according to the tag description. The complete list of mappings is published in Appendix A.

Experimental setup and results

Settings The feature representations are trained on the union of the PTB and the PPCEME. The domain attributes for FEMA are set to include the three corpora themselves (PTB, PPCMBE, and PPCEME), and the genre attributes in the historical corpora. Note that all sentences in the Penn Treebank WSJ data belong to the same genre (news). We also consider the well-known Brown clustering method [123] as another baseline. For SCL, we use the same threshold of 50 occurrences for pivot features, and include 8089 features that pass this threshold. PTB WSJ sections 19-21 are used for parameter tuning: we find that the best number of Brown clusters is 200, and the optimum embedding sizes are 200 and 100 for word2vec and FEMA.

Spelling normalization Spelling variants lead to a high percentage of out-of-vocabulary (OOV) tokens in historical texts, which poses problems for POS tagging. We normalize the PPCEME sentences using VARD [28], a widely used spelling normalization tool that has been proven to improve performance on POS tagging [10] and syntactic parsing [124]. VARD is designed specifically for Early Modern English spelling variation, and additional labeled data and training are required for other forms of spelling variation, which we do not consider here. Following Schneider *et al.* [124], we utilize VARD's auto-normalization function with a 50% normalization threshold, achieving a balance between precision and

Table 4.8: Accuracy results for adapting from the PTB to the PPCMBE and the PPCEME of historical English.

Target	Normalized	baseline		SCL	Brown	word2vec	FEMA	
		SVM	MEMM (Stanford)				single embedding	attribute embeddings
PPCMBE	No	81.12	81.35	81.66	81.65	81.75	82.34	**82.46**
PPCEME	No	74.15	74.34	75.89	76.04	75.85	77.77	**77.92**
PPCEME	Yes	76.73	76.87	77.61	77.65	77.76	78.85	**79.05**

recall. At this threshold, a total of 12% ($236298/1961157$) of the tokens in the PPCEME are normalized.[6]

Results As shown in Table 4.8, this task is considerably difficult, with even the best systems achieving accuracies that are nearly 15% worse than in-domain training. Nonetheless, domain adaptation can help: FEMA improves performance by 1.3% on the PPCMBE data, and by 3.8% on the unnormalized PPCEME data. Spelling normalization also helps, improving the baseline systems by more than 2.5%. The combination of spelling normalization and domain adaptation gives an overall improvement in accuracy from 74.2% to 79.1%.

Analysis

As expected, the Early Modern English dataset (PPCEME) is considerably more challenging than the Modern British English dataset (PPCMBE): the baseline accuracy is 7% worse on the PPCEME than the PPCMBE. However, the PPCEME is also more amenable to domain adaptation, with FEMA offering considerably larger improvements. One reason is that the PPCEME has many more out-of-vocabulary (OOV) tokens: 23%, versus 9.2% in the PPCMBE. Both domain adaptation and normalization help to address this specific issue, and they yield further improvements when used in combination. This subsection offers further insights on the sources of errors and possibilities for improvement on the PPCEME

[6]We only consider $1:1$ mappings, and ignore 328 normalizations corresponding to $1:n$ mappings.

Table 4.9: Tagging accuracies of adaptation of our baseline SVM tagger from the PTB to the PPCEME in ablation experiments.

Feature set	IV	OOV	All
All features	81.68	48.96	74.15
– word context	79.69	38.62	70.23
– prefix	81.61	46.11	73.43
– suffix	81.36	38.13	71.40
– affix	81.22	34.40	70.44
– orthographic	81.68	48.92	74.14

data.

Feature Ablation Table 4.9 presents the results of feature ablation experiments for the non-adapted SVM tagger. Word context features are important for obtaining good accuracies on both IV and OOV tokens. Affix features, particularly suffix features, are crucial for the OOV tokens. The orthographic features are shown to be nearly irrelevant, as long as affix features are present. Overall, the high percentage of OOV tokens can be a major source of errors, as the tagging accuracy on OOV tokens is below 50% in our best baseline system. Note that these results are for a classification-based tagger; while the Viterbi-based MEMM tagger performs only marginally better overall ($\sim 0.2\%$ improvement), it is possible that its error distribution might be different due to the advantages of structured prediction.

Error Analysis The accuracy on out-of-vocabulary (OOV) tokens is generally low, and spelling variation is a major source of OOV tokens. For instance, 'ye' and 'thy', the older forms of 'the' and 'your', are often incorrectly tagged as `NN` and `JJ` in the PPCEME. In general, the per-tag accuracies are roughly correlated with the percentages of OOV tokens. Some exceptions including `VB`, `NNP` and `NNS`, where the affix features can be very useful for tagging OOV tokens.

That said, the cross-domain accuracy on in-vocabulary (IV) tokens is also low, at roughly 80% when adapting from the PTB to the PPCEME. A major source of error here

is the mismatch in annotation schemes between the two datasets, which is only partially addressed by a deterministic tag mapping. Table 4.10 presents the SVM accuracy per tag, and the most common error correspondingly. Most of the errors shown in the table are owing to different annotations of the same token in the two corpora.

One major cause of errors is in misalignments of punctuations and their POS tags. For example, in the PPCEME, 16.6% of commas are labeled as `.` (sentence-final punctuation), and 12.3% periods are labeled as `,` (sentence-internal punctuation); these punctuations are less ambiguous in the PTB. The historical corpora lack special tags for colons and ellipses, which are present in the PTB. In contrast to the PTB, there is no distinction between opening quotation mark and closing quotation mark in the PPCEME. Moon and Baldridge [120] avoid these difficulties by mapping all the punctuation tokens to a single tag. We did not follow their setting because it would lead to a significant change of test data. However, it should be noted that these "errors" are not particularly meaningful for linguistic analysis, and could easily be addressed by heuristic post-processing.

The tagging performance is also impaired by the different annotations of many common words. For example, in the PTB, more than 99.9% of token 'to' are labeled as `TO`, but in the PCHE this word can also be labeled as `IN`, distinguishing the infinitive marker from the preposition. The words 'all', 'any' and 'every' are annotated as quantifiers in the PCHE; this tag is mapped to `JJ`, but these specific words are all labeled as `DT` in the PTB. A simple remapping from `Q` to `DT` leads to an increase of 0.78% baseline accuracy; it is possible that other changes to the tag mappings of Moon and Baldridge [120] might yield further improvements, but a more systematic approach would be outside the bounds of *unsupervised* domain adaptation.

Improvements from Normalization As shown above, the tagging accuracy decreases from 81.7% on IV tokens to 49.0% on OOV tokens. Spelling normalization helps to increase the accuracy by transforming OOV tokens to IV tokens. After normalization, the

Table 4.10: Accuracy (recall) rates per tag with the SVM model, for the 15 most common tags. For each gold category, the most common error word and predicted tag are shown.

Tag	% of OOV	Accuracy	Most common error
IN	6.93	82.79	to/TO
NN	48.39	64.74	Lord/NNP
DT	3.45	94.62	that/IN
PRP	13.57	78.80	other/JJ
,	0.41	87.86	./.
JJ	32.20	48.60	all/DT
CC	1.98	91.29	for/IN
RB	26.22	65.74	such/JJ
.	0.56	54.43	,/,
VB	34.69	75.06	have/VBP
NNP	58.91	88.31	god/NN
NNS	59.12	73.88	Lords/NNPS
VBD	25.87	81.93	quoth/NN
VBN	37.75	63.09	said/VBD
PRP$	13.57	85.49	thy/JJ

OOV rate for the PPCEME falls from 23.0% to 13.5%, corresponding to a reduction of 41.5% OOV tokens. Normalization is not perfectly accurate, and the tagging performance for IV tokens drops slightly to 81.2% on IV tokens. But due to the dramatic decrease in the number of OOV tokens, normalization improves the overall accuracy by more than 2.5%. We also observe performance drops on tagging OOV tokens after normalization (49.0% to 48.1%), which suggests that the remaining unnormalized OOV tokens are the tough cases for both normalization and POS tagging.

Improvements from Domain Adaptation As presented in Table 4.11, the tagging accuracies are increased on both IV and OOV tokens with the domain adaptation methods. Compared against the baseline tagger, FEMA-attribute achieves an absolute improvement of 14% in accuracy on OOV tokens. SCL performs slightly better than Brown clustering and word2vec on IV tokens, but worse on OOV tokens. By incorporating metadata attributes, FEMA-attribute performs better than FEMA-single on OOV tokens, though the accuracies on IV tokens are similar. Interestingly, the venerable method of Brown clustering (slightly) outperforms both word2vec and SCL.

Table 4.11: Tagging accuracies of domain adaptation models from the PTB to the PPCEME.

System	IV	OOV	All
SVM	81.68	48.96	74.15
SCL	82.01	55.45	75.89
Brown	81.81	56.76	76.04
word2vec	81.79	56.00	75.85
FEMA-single	82.30	62.63	77.77
FEMA-attribute	82.34	63.16	77.92

We further study the relationship between domain adaptation and spelling normalization by looking into the errors corrected by both approaches. Domain adaptation yields larger improvements than spelling normalization on both IV and OOV tokens, although as noted above, the approaches are somewhat complementary. The results show that among the 60,928 error tokens corrected by VARD, 60% are also corrected by FEMA-attribute, while the remaining 40% would be left uncorrected by the domain adaptation technique. Conversely, among the errors corrected by FEMA-attribute, 38% are also corrected by VARD, while the remaining 62% would be left uncorrected. The overlap of reduced errors is because both approaches exploit similar sources of information, including affixes and local word contexts.

4.4 Similarity in the Embedding Space

The utility of word and feature embeddings for POS tagging task can be evaluated through word similarity in the embedding space, and its relationship to type-level part-of-speech labels. To measure the label consistency between each word and its top Q closest words in the vocabulary we compute,

$$\text{Consistency} = \frac{\sum_{i=1}^{|V|}\sum_{j=1}^{Q}\beta(w_i, c_{ij})}{|V| \times Q} \tag{4.6}$$

Table 4.12: Label consistency of the Q-most similar words in each embedding. FEMA-all is the concatenation of the current, previous, and next-word FEMA embeddings.

Embedding	$Q = 5$	$Q = 10$	$Q = 50$	$Q = 100$
WORD2VEC	47.64	46.17	41.96	40.09
FEMA-current	68.54	66.93	62.36	59.94
FEMA-prev	55.34	54.18	50.41	48.39
FEMA-next	57.13	55.78	52.04	49.97
FEMA-all	70.63	69.60	65.95	63.91

where $|V|$ is the number of words in the vocabulary, w_i is the i-th word in the vocabulary, c_{ij} is the j-th closest word to w_i in the embedding space (using cosine similarity), $\beta(w_i, c_{ij})$ is an indicator function that is equal to 1 if w_i and c_{ij} have the same most common part-of-speech in labeled data.

We compare feature embeddings of different templates against WORD2VEC embeddings. All embeddings are trained on the SANCL data, which is also used to obtain the most common tag for each word. Table 4.12 shows that the FEMA embeddings are more consistent with the type-level POS tags than WORD2VEC embeddings. This is not surprising, since they are based on feature templates that are specifically designed for capturing syntactic regularities. In simultaneously published work, Ling *et al.* [125] present "position-specific" word embeddings, which are an alternative method to induce more syntactically-oriented word embeddings.

Table 4.13 shows the most similar words for three query keywords, in each of four different embeddings. The next-word and previous-word embeddings are most related to syntax, because they help to predict each other and the current-word feature; the current-word embedding brings in aspects of orthography, because it must help to predict the affix features. In morphologically rich languages such as Portuguese, this can help to compute good embeddings for rare inflected words. This advantage holds even in English: the word 'toughness' appears only once in the SANCL data, but the FEMA-current embedding is able to capture its morphological similarity to words such as 'tightness' and 'thickness'. In WORD2VEC, the lists of most similar words tend to combine syntax and topic information,

Table 4.13: Most similar words for three queries, in each embedding space.

'**new**'	
FEMA-current	nephew, news, newlywed, newer, newspaper
FEMA-prev	current, local, existing, international, entire
FEMA-next	real, big, basic, local, personal
WORD2VEC	current, special, existing, newly, own
'**toughness**'	
FEMA-current	tightness, trespass, topless, thickness, tenderness
FEMA-prev	underside, firepower, buzzwords, confiscation, explorers
FEMA-next	aspirations, anguish, pointers, organisation, responsibilities
WORD2VEC	parenting, empathy, ailment, rote, nerves
'**and**'	
FEMA-current	amd, announced, afnd, anesthetized, anguished
FEMA-prev	or, but, as, when, although
FEMA-next	or, but, without, since, when
WORD2VEC	but, while, which, because, practically

and fail to capture syntactic regularities such as the relationship between 'and' and 'or'.

4.5 Related Work

Multi-domain adaptation The question of adaptation across multiple domains has mainly been addressed in the context of supervised multi-domain learning, with labeled data available in all domains [126]. Finkel and Manning [74] propagate classification parameters across a tree of domains, so that classifiers for sibling domains are more similar; Daumé III [73] shows how to induce such trees using a nonparametric Bayesian model. Dredze *et al.* [127] combine classifier weights using confidence-weighted learning, which represents the covariance of the weight vectors. Joshi *et al.* [108] formulate the problem of multi-attribute multi-domain learning, where all attributes are potential distinctions between domains; Want *et al.* [128] present an approach for automatically partitioning instances into domains according to such metadata features. Our formulation is related to multi-domain learning, particularly in the multi-attribute setting. However, rather than partitioning all instances into domains, the domain attribute formulation allows information to be shared across instances which share metadata attributes. We are unaware of prior research on

unsupervised multi-domain adaptation.

Word embeddings Word embeddings can be viewed as special case of representation learning, where the goal is to learn representations for each word, and then to supply these representations in place of lexical features. Early work focused on discrete clusters [129], while more recent approaches induce dense vector representations; Turian *et al.* [103] compare Brown clusters with neural word embeddings from Collobert and Weston [130] and Mnih and Hinton [131]. Word embeddings can also be computed via neural language models [114], or from canonical correlation analysis [132]. Xiao and Guo [104] induce word embeddings across multiple domains, and concatenate these representations into a single feature vector for labeled instances in each domain, following EasyAdapt [126]. However, they do not apply this idea to unsupervised domain adaptation, and do not work in the structured feature setting that we consider here. Bamman *et al.* [133] learn geographically-specific word embeddings, in an approach that is similar to our multi-domain feature embeddings, but they do not consider the application to domain adaptation. We can also view the distributed representations in FLORS as a sort of word embedding, computed directly from rescaled bigram counts [110].

Feature embeddings are based on a different philosophy than word embeddings. While many NLP features are lexical in nature, the role of a word towards linguistic structure prediction may differ across feature templates. Applying a single word representation across all templates is therefore suboptimal. Another difference is that feature embeddings can apply to units other than words, such as character strings and shape features. The tradeoff is that feature embeddings must be recomputed for each set of feature templates, unlike word embeddings, which can simply be downloaded and plugged into any NLP problem. However, computing feature embeddings is easy in practice, since it requires only a light modification to existing well-optimized implementations for computing word embeddings.

Historical texts Historical texts differ from modern texts in spellings, syntax and semantics, posing significant challenges for standard NLP systems, which are usually trained with modern news text. Numerous resources have been created for overcoming the difficulties, including syntactically annotated corpora [99, 100, 29] and spelling normalization tools [4, 28]. Most previous work focuses on normalization, which can significantly increase tagging accuracy on historical English [10] and German [134]. Similar improvements have been obtained for syntactic parsing [124]. Domain adaptation offers an alternative approach which is more generic—for example, it can be applied to any corpus without requiring the design of a set of normalization rules. As shown above, when normalization is possible, it can be combined with domain adaptation to yield better performance than that obtained by either approach alone.

CHAPTER 5

SOCIALLY ADAPTED NATURAL LANGUAGE PROCESSING

In this chapter we propose to overcome language variation by leveraging social network structures that are available in many online social networking sites. In particular, we explore the sociological theory of *homophily*, which asserts that socially connected individuals are more likely to have similar behaviors or share similar interests [34]. This property has been demonstrated both for language [35, 36] as well as for the demographic properties targeted by Hovy [30], which are more likely to be shared by friends than by random pairs of individuals [37, 38]. We illustrate that the social theory can be used to improve two natural language processing tasks: microblog entity linking and sentiment analysis.

One challenge of applying NLP systems on social media texts is that we often lack of sufficient local context to tackle the pervasiveness of ambiguity in natural languages. For example, as shown in Figure 1.6, the entity mention 'Giants' in tweet t_1 can refer to the NFL football team *New York Giants* or the MLB baseball team *San Francisco Giants*. In this example, it is impossible to disambiguate between these entities solely based on the individual text message. We assume Twitter users will have similar interests in real world entities to their near neighbors—an assumption of *entity homophily*. The social relation between users u_1 and u_2 may lead to more coherent topics in tweets t_1 and t_2. Therefore, by successfully linking the less ambiguous mention 'Red Sox' in tweet t_2 to the *Boston Red Sox* baseball team, the tweet entity linking system will be more confident on linking 'Giants' to the *San Francisco Giants* football team in tweet t_1. In section 5.1, we show that this assumption can be used to improve the entity disambiguation capability of the entity linking system.

Another challenge is that words can mean different things to different people. For example, the word 'sick' typically has a negative sentiment, e.g., 'I would like to believe he's

sick rather than just mean and evil'.[1] However, in some communities the word can have a positive sentiment, e.g., the lyric 'this sick beat', recently trademarked by the musician Taylor Swift. Given labeled examples of 'sick' in use by individuals in a social network, we assume that the word will have a similar sentiment meaning for their near neighbors—an assumption of *linguistic homophily*. Note that this differs from the assumption of *label homophily*, which entails that neighbors in the network will hold similar opinions, and will therefore produce similar document-level labels [22, 39, 40]. Linguistic homophily is a more generalizable claim, which could in principle be applied to any language processing task where author network information is available. In section 5.2, we show that this assumption can be used to improve sentiment analysis on custom reviews and Twitter messages.

5.1 Socially-Infused Information Extraction

Entity linking on short texts (e.g., Twitter messages) is of increasing interest, as it is an essential step for many downstream applications, such as market research [17], topic detection and tracking [135], and question answering [136]. However, tweet entity linking is a particularly difficult problem, because the short context around an entity mention is often insufficient for entity disambiguation. To address this problem, we propose to apply the assumption of entity homophily and employ social network structures as additional context information for disambiguating entities.

Specifically, we adopt the recent advance on embedding information networks [137], which induces low-dimensional representations for author nodes based on the network structure. By learning the semantic interactions between the author embeddings and the pre-trained Freebase entity embeddings, the entity linking system can incorporate more disambiguating context from the social network. We also consider low-dimensional representations of mentions, another source of related information for entity linking, with the

[1] Charles Rangel, describing Dick Cheney

Table 5.1: Statistics of data sets.

Data	# Tweet	# Entity	Date
NEEL-train	2,340	2,202	Jul. - Aug. 2011
NEEL-test	1,164	687	Jul. - Aug. 2011
TACL	500	300	Dec. 2012

intuition that semantically related mentions can refer to similar entities.

Previously proposed approaches [138, 139] are based on hand-crafted features and off-the-shelf machine learning algorithms. Our preliminary study suggests that simply augmenting the traditional surface features with the distributed representations barely improves the performance of these entity linking systems. Therefore, we propose NTEL, a **N**eural model for **T**weet **E**ntity **L**inking, to leverage the distributed representations of authors, mentions, and entities. NTEL can not only make efficient use of statistical surface features built from a knowledge base, but also learn the interactions between these distributed representations. We perform message-level inference using a dynamic program to avoid overlapping mentions. The architecture is trained with loss-augmented decoding, a large margin learning technique for structured prediction. The complete system, NTEL, outperforms the previous state-of-the-art [139] by 3% average F1 on two benchmark datasets.

5.1.1 Data

Two publicly available datasets for tweet entity linking are adopted in the work. NEEL is originally collected and annotated for the Named Entity Extraction & Linking Challenge [140], and TACL is first used and released by Fang and Chang [141]. The datasets are then cleaned and unified by Yang and Chang [139]. The statistics of the datasets are presented in Table 5.1.

5.1.2 Testing entity homophily

The hypothesis of *entity homophily*, as presented in the introduction, is that socially connected individuals are more likely to mention similar entities than disconnected individuals.

Table 5.2: The average entity-driven similarity results for the networks.

Network	$sim(i \leftrightarrow j)$	$sim(i \not\leftrightarrow j)$
FOLLOWER	0.128	0.025
MENTION	0.121	0.025
RETWEET	0.173	0.025

We now test the hypothesis on real data before we start building our entity linking systems.

Twitter social networks We test the assumption on the users in the NEEL-train dataset. We construct three author social networks based on the follower, mention and retweet relations between the 1,317 authors in the NEEL-train dataset, which we refer as FOLLOWER, MENTION and RETWEET. Specifically, we use the Twitter API to crawl the friends of the NEEL users (individuals that they follow) and the mention/retweet links are induced from their most recent 3,200 tweets.[2] We exploit bi-directed links to create the undirected networks, as bi-directed links result in stronger social network ties than directed links [142, 143]. The numbers of social relations for the networks are 1,604, 379 and 342 respectively.

Metrics We propose to use the *entity-driven similarity* between authors to test the hypothesis of entity homophily. For a user u_i, we employ a Twitter NER system [12] to detect entity mentions in the timeline, which we use to construct a user entity vector $\mathbf{u}_i^{(ent)}$, so that $u_{i,j}^{(ent)} = 1$ iff user i has mentioned entity j.[3] The entity-driven similarity between two users u_i and u_j is defined as the cosine similarity score between the vectors $\mathbf{u}_i^{(ent)}$ and $\mathbf{u}_j^{(ent)}$. We evaluate the three networks by calculating the average entity-driven similarity of the connected user pairs and that of the disconnected user pairs, which we name as $sim(i \leftrightarrow j)$ and $sim(i \not\leftrightarrow j)$.

Results The entity-driven similarity results of these networks are presented in Table 5.2. As shown, $sim(i \leftrightarrow j)$ is substantially higher than $sim(i \not\leftrightarrow j)$ on all three social networks,

[2]We are able to obtain at most 3,200 tweets for each Twitter user, due to the Twitter API limits.

[3]We assume each name corresponds to a single entity for this metric, so this metric only approximates entity homophily.

indicating that socially connected individuals clearly tend to mention more similar entities than disconnected individuals. Note that $sim(i \not\leftrightarrow j)$ is approximately equal to the same base rate defined by the average entity-driven similarity of all pairs of users, because the vast majority of user pairs are disconnected, no matter how to define the network. Among the three networks, RETWEET offers slightly higher $sim(i \leftrightarrow j)$ than FOLLOWER and MENTION. The results verify our hypothesis of entity homophily, which forms the basis for this research. Note that all social relation data was acquired in March 2016; by this time, the authorship information of 22.1% of the tweets in the NEEL-train dataset was no longer available, because the tweets or user accounts had been deleted.

5.1.3 Method

In this section, we present, NTEL, a novel neural based tweet entity linking framework that is able to leverage social information. We first formally define the task of tweet entity linking. Assume we are given an entity database (e.g., Wikipedia or Freebase), and a lexicon that maps a surface form into a set of entity candidates. For each input tweet, we consider any n-grams of the tweet that match the lexicon as mention candidates.[4] The entity linking system maps every mention candidate (e.g., 'Red Sox') in the message to an entity (e.g., *Boston Red Sox*) or to **Nil** (i.e., not an entity). There are two main challenges in the problem. First, a mention candidate can often potentially link to multiple entities according to the lexicon. Second, as shown in Figure 5.1, many mention candidates overlap with each other. Therefore, the entity linking system is required to disambiguate entities and produce non-overlapping entity assignments with respect to the mention candidates in the tweet.

We formalize this task as a structured learning problem. Let $\mathbf{x}$ be the tweet, u be the author, and $\mathbf{y} = \{y_t\}_{t=1}^{T}$ be the entity assignments of the T mention candidates in the tweet.

[4]We adopted the same entity database and lexicon as those used by Yang and Chang [139].

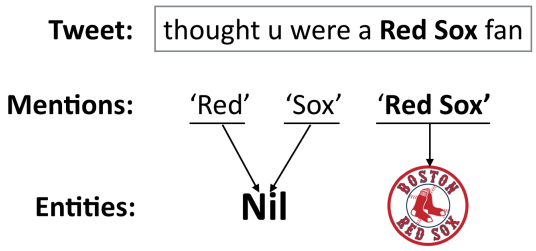


Figure 5.1: Illustration of the non-overlapping structure for the task of tweet entity linking. In order to link 'Red Sox' to a real entity, 'Red' and 'Sox' should be linked to **Nil**.

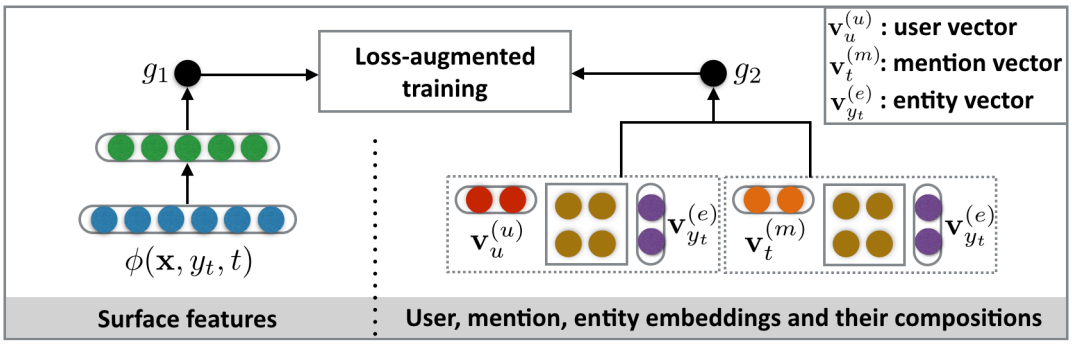


Figure 5.2: The proposed neural network approach for tweet entity linking. A composition model based on bilinear functions is used to learn the semantic interactions of user, mention, and entity.

The overall scoring function $s(\mathbf{x}, \mathbf{y}, u)$ can be decomposed as follows,

$$s(\mathbf{x}, \mathbf{y}, u) = \sum_{t=1}^{T} g(\mathbf{x}, y_t, u, t), \tag{5.1}$$

where $g(\mathbf{x}, y_t, u, t)$ is the scoring function for the t-th mention candidate choosing entity y_t. Note that the system needs to produce non-overlapping entity assignments, which will be resolved in the inference algorithm.

The overview of NTEL is illustrated in Figure 5.2. A small number of hand-crafted features are important for the task due to the small training data [139]. As the surface features are very different from the distributed features, we further break down $g(\mathbf{x}, y_t, u, t)$ into two scoring functions to deal with the two types of representations separately:

$$g(\mathbf{x}, y_t, u, t; \Theta_1, \Theta_2) = g_1(\mathbf{x}, y_t, t; \Theta_1) + g_2(\mathbf{x}, y_t, u, t; \Theta_2), \tag{5.2}$$

where g_1 is the scoring function for our basic surface features, and g_2 is the scoring function for modeling user, mention, entity representations and their compositions. Θ_1 and Θ_2 are model parameters that will be detailed below. We choose to use a multilayer perceptron (MLP) to model $g_1(\mathbf{x}, y_t, t; \Theta_1)$, and we employ simple yet efficient bilinear functions to learn the compositions of user, mention, and entity representations $g_2(\mathbf{x}, y_t, u, t; \Theta_2)$. Finally, we present a training algorithm based on loss-augmented decoding and a non-overlapping inference algorithm.

Modeling surface features

We include the 37 features used by Yang and Chang [139] as our surface feature set. These features are extracted from various sources, including a named entity recognizer, an entity type recognizer, and some statistics of the Wikipedia pages.

We exploit a multilayer perceptron (MLP) to transform the surface features to a real-valued score. The output of the MLP is formalized as follows,

$$\begin{aligned} g_1(\mathbf{x}, y_t, t; \Theta_1) =& \boldsymbol{\beta}^\top \mathbf{h} + b \\ \mathbf{h} =& \tanh(\mathbf{W}\phi(\mathbf{x}, y_t, t) + \mathbf{b}), \end{aligned} \tag{5.3}$$

where $\phi(\mathbf{x}, y_t, t)$ is the feature function, $\mathbf{W}$ is an $M \times D$ matrix, the weights $\mathbf{b}$ are bias terms, and $\mathbf{h}$ is the output of the hidden layer of the MLP. $\boldsymbol{\beta}$ is an M dimensional vector of weights for the output score, and b is the bias term. The parameters of the MLP are $\Theta_1 = \{\mathbf{W}, \mathbf{b}, \boldsymbol{\beta}, b\}$. Yang and Chang [139] argue that non-linearity is the key for obtaining good results on the task, as linear models are not expressive enough to capture the high-order relationships between the dense features. They propose a tree-based non-linear model for the task. The MLP forms simple non-linear mappings between the input features and the output score, whose parameters will be jointly learnt with other components in NTEL.

Modeling user, mention, and entity

To leverage the social network structure, we first train low-dimensional embeddings for the authors using the social relations. The mention and entity representations are given by word embeddings learnt with a large Twitter corpus and pre-trained Freebase entity embeddings respectively. We will denote the user, word, entity embedding matrices as:

$$\mathbf{E}^{(u)} = \{\mathbf{v}_u^{(u)}\} \qquad \mathbf{E}^{(w)} = \{\mathbf{v}_w^{(w)}\} \qquad \mathbf{E}^{(e)} = \{\mathbf{v}_e^{(e)}\},$$

where $\mathbf{E}^{(u)}, \mathbf{E}^{(w)}, \mathbf{E}^{(e)}$ are $V^{(u)} \times D^{(u)}, V^{(w)} \times D^{(w)}, V^{(e)} \times D^{(e)}$ matrices, and $\mathbf{v}_u^{(u)}$, $\mathbf{v}_w^{(w)}$, $\mathbf{v}_e^{(e)}$ are $D^{(u)}$, $D^{(w)}$, $D^{(e)}$ dimensional embedding vectors respectively. $V^{(u)}, V^{(w)}, V^{(e)}$ are the vocabulary sizes for users, words, and entities. Finally, we present a composition model for learning semantic interactions between user, mention, and entity.

User embeddings We obtain low-dimensional Twitter author embeddings $\mathbf{E}^{(u)}$ using *LINE*—the recently proposed model for embedding information networks [137]. Specifically, we train *LINE* with the second-order proximity, which assumes that Twitter users sharing many neighbors are close to each other in the embedding space. According to the original paper, the second-order proximity yields slightly better performance than the first-order proximity, which assumes connecting users are close to each other, on a variety of downstream tasks.

Mention embeddings The representation of a mention is the average of embeddings of words it contains. As each mention is typically one to three words, the simple representations often perform surprisingly well [144]. We adopt the structured skip-gram model [125] to learn the word embeddings $\mathbf{E}^{(w)}$ on a Twitter corpus with 52 million tweets [60]. The

mention vector of the t-th mention candidate can be written as:

$$\mathbf{v}_t^{(m)} = \frac{1}{|\mathbf{x}_t^{(w)}|} \sum_{w \in \mathbf{x}_t^{(w)}} \mathbf{v}_w^{(w)}, \tag{5.4}$$

where $\mathbf{x}_t^{(w)}$ is the set of words in the mention.

Entity embeddings We use the pre-trained Freebase entity embeddings released by Google to represent entity candidates, which we refer as $\mathbf{E}^{(e)}$.[5] The embeddings are trained with the skip-gram model [114] on 100 billion words from various news articles. The entity embeddings can also be learnt from Wikipedia hyperlinks or Freebase entity relations, which we leave as future work.

Compositions of user, mention, and entity The distributed representations of users, mentions, and entities offer additional information that is useful for improving entity disambiguation capability. In particular, we explore the information by making two assumptions: socially connected users are interested in similar entities (entity homophily), and semantically related mentions are likely to be linked to similar entities.

We utilize a simple composition model that takes the form of the summation of two bilinear scoring functions, each of which explicitly leverages one of the assumptions.[6] Given the author representation $\mathbf{v}_u^{(u)}$, the mention representation $\mathbf{v}_t^{(m)}$, and the entity representation $\mathbf{v}_{y_t}^{(e)}$, the output of the model can be written as:

$$g_2(\mathbf{x}, y_t, u, t; \Theta_2) = {\mathbf{v}_u^{(u)}}^\top \mathbf{W}^{(u,e)} \mathbf{v}_{y_t}^{(e)} + {\mathbf{v}_t^{(m)}}^\top \mathbf{W}^{(m,e)} \mathbf{v}_{y_t}^{(e)}, \tag{5.5}$$

where $\mathbf{W}^{(u,e)}$ and $\mathbf{W}^{(m,e)}$ are $D^{(u)} \times D^{(e)}$ and $D^{(w)} \times D^{(e)}$ bilinear transformation matrices. Similar bilinear formulation has been used in the literature of knowledge base com-

[5] Available at https://code.google.com/archive/p/word2vec/

[6] The summation of two inner product functions is a simpler form, but it is inappropriate here due to different lengths of the distributed representations.

pletion and inference [144, 145]. The parameters of the composition model are $\Theta_2 = \{\mathbf{W}^{(u,e)}, \mathbf{W}^{(m,e)}, \mathbf{E}^{(u)}, \mathbf{E}^{(w)}, \mathbf{E}^{(e)}\}$.

Non-overlapping inference

The non-overlapping constraint for entity assignments requires inference method that is different from the standard Viterbi algorithm for a linear chain. We now present a variant of the Viterbi algorithm for the non-overlapping structure. Given the overall scoring function $g(\mathbf{x}, y_t, u, t)$ for the t-th mention candidate choosing an entity y_t, we sort the mention candidates by their end indices and define the Viterbi recursion by

$$\hat{y}_t = {}_{y_t \in \mathcal{Y}_{\mathbf{x}_t}, y_t \neq \textbf{Nil}} g(\mathbf{x}, y_t, u, t) \tag{5.6}$$

$$a(1) = \max(g(\mathbf{x}, \textbf{Nil}, u, 1), g(\mathbf{x}, \hat{y}_1, u, 1)) \tag{5.7}$$

$$a(t) = \max\left(\psi_t(\textbf{Nil}), \psi_t(\hat{y}_t)\right) \tag{5.8}$$

$$\psi_t(\textbf{Nil}) = g(\mathbf{x}, \textbf{Nil}, u, t) + a(t-1) \tag{5.9}$$

$$\psi_t(\hat{y}_t) = g(\mathbf{x}, \hat{y}_t, u, t) + \sum_{prev(t) < t' < t} g(\mathbf{x}, \textbf{Nil}, u, t') + a(prev(t)) \tag{5.10}$$

where $\mathcal{Y}_{\mathbf{x}_t}$ is set of entity candidates for the t-th mention candidate, and $prev(t)$ is a function that points out the previous non-overlapping mention candidate for the t-th mention candidate. We exclude any second-order features between entities. Therefore, for each mention candidate, we only need to decide whether it can take the highest scored entity candidate $\hat{y}_t$ or the special **Nil** entity based on whether it is overlapped with other mention candidates.

Loss-augmented training

The parameters need to be learnt during training are $\Theta = [\Theta_1, \{\mathbf{W}^{(u,e)}, \mathbf{W}^{(m,e)}\}]$.[7] We train NTEL by minimizing the following loss function for each training tweet:

$$L(\Theta) = \max_{\mathbf{y} \in \mathcal{Y}_\mathbf{x}} (\Delta(\mathbf{y}, \mathbf{y}^*) + s(\mathbf{x}, \mathbf{y}, u)) - s(\mathbf{x}, \mathbf{y}^*, u), \tag{5.11}$$

where $\mathbf{y}^*$ is the gold structure, $\mathcal{Y}_\mathbf{x}$ represents the set of valid output structures for $\mathbf{x}$, and $\Delta(\mathbf{y}, \mathbf{y}^*)$ is the weighted hamming distance between the gold structure $\mathbf{y}^*$ and the valid structure $\mathbf{y}$. The hamming loss is decomposable on the mention candidates, which enables efficient inferences. We set the hamming loss weight to 0.2 after a preliminary search. Note that the number of parameters in our composition model is large. Thus, we include an L2 regularizer on these parameters, which is omitted from Equation 5.11 for brevity. The evaluation of the loss function corresponds to the loss-augmented inference problem:

$$\hat{\mathbf{y}} =_{\mathbf{y} \in \mathcal{Y}_\mathbf{x}} (\Delta(\mathbf{y}, \mathbf{y}^*) + s(\mathbf{x}, \mathbf{y}, u)), \tag{5.12}$$

which can be solved by the above non-overlapping inference algorithm. We employ vanilla SGD algorithm to optimize all the parameters. The numbers of training epochs are determined by early stopping (at most 1000 epochs). Training takes 6-8 hours on 4 threads.

5.1.4 Experiments

In this section, we evaluate NTEL on the NEEL and TACL datasets as described in subsection 5.1.1, focusing on investigating whether social information can improve the task. We also compare NTEL with the previous state-of-the-art system.

[7] We fixed the pre-trained embedding matrices during loss-augmented training.

Table 5.3: Statistics of author social networks used for training user embeddings.

Network	# Author	# Relation
FOLLOWER+	8,772	286,800
MENTION+	6,119	57,045
RETWEET+	7,404	59,313

Social network expansion

We utilize Twitter follower, mention, and retweet social networks to train user embeddings. We were able to identify 2,312 authors for the tweets of the two datasets in March 2016. We then used the Twitter API to crawl their friend links and timelines, from which we can induce the networks. We find the numbers of social connections (bidirectional links) between these users are relatively small. In order to learn better user embeddings, we expand the set of author nodes by including nodes that will do the most to densify the author networks. For the follower network, we add additional individuals who are followed by at least twenty authors in the original set. For the mention or retweet networks, we add all users who have mentioned or retweeted by at least ten authors in the original set. The statistics of the resulting networks are presented in Table 5.3.

Experimental settings

Following Yang and Chang [139], we train all the models with the NEEL-train dataset and evaluate different systems on the NEEL-test and TACL datasets. In addition, 800 tweets from the NEEL-train dataset are sampled as our development set to perform parameter tuning. Note that Yang and Chang [139] also attempt to optimize F1 scores by balancing precision and recall scores on the development set; we do not fine tune our F1 in this way, so that we can apply a single trained system across different test sets.

Metrics We follow prior work [138, 139] and perform the standard evaluation for an end-to-end entity linking system, computing precision, recall, and F1 score according to

the entity references and the system outputs. An output entity is considered as correct if it matches the gold entity and the mention boundary overlaps with the gold mention boundary. More details about the metrics are described by Carmel *et al.* [146].

Competitive systems Our first baseline system, NTEL-nonstruct, ignores the structure information and makes the entity assignment decision for each mention candidate individually. For NTEL, we start with a baseline system using the surface features, and then incorporate the two bilinear functions (user-entity and mention-entity) described in Equation 5.5 incrementally. Our main evaluation uses the RETWEET+ network, since the retweet network had the greatest entity homophily; an additional evaluation compares across network types.

Parameter tuning We tune all the hyper-parameters on the development set, and then retrain the models on the full training data with the best parameters. We choose the number of hidden units for the MLP from $\{20, 30, 40, 50\}$, and the regularization penalty for our composition model from $\{0.001, 0.005, 0.01, 0.05, 0.1\}$. The sizes of user embeddings and word embeddings are selected from $\{50, 100\}$ and $\{200, 400, 600\}$ respectively. The pre-trained Freebase entity embedding size is 1000. The learning rate for the SGD algorithm is set as 0.01. During training, we check the performance on the development set regularly to perform early stopping.

Results

Table 5.4 summarizes the empirical findings for our approach and S-MART [139] on the tweet entity linking task. For the systems with user-entity bilinear function, we report results obtained from embeddings trained on RETWEET+ in Table 5.4, and other results are available in Table 5.5. The best hyper-parameters are: the number of hidden units for the MLP is 40, the L2 regularization penalty for the composition parameters is 0.005, and the user embedding size is 100. For the word embedding size, we find 600 offers marginal

Table 5.4: Evaluation results on the NEEL-test and TACL datasets for different systems. The best results are in **bold**.

System	user -entity	mention -entity	NEEL-test			TACL			Avg. F1
			P	R	F1	P	R	F1	
Our approach									
NTEL-nonstruct			80.0	68.0	73.5	64.7	62.3	63.5	68.5
NTEL			**82.8**	69.3	75.4	68.0	66.0	67.0	71.2
NTEL	✓		82.3	71.8	76.7	66.9	68.7	67.8	72.2
NTEL		✓	80.2	**75.8**	77.9	66.9	**69.3**	68.1	73.0
NTEL	✓	✓	81.9	75.6	**78.6**	**69.0**	69.0	**69.0**	**73.8**
Best published results									
S-MART			80.2	75.4	77.7	60.1	67.7	63.6	70.7

Table 5.5: Comparison of different social networks with our full model. The best results are in **bold**.

Network	NEEL-test			TACL		
	P	R	F1	P	R	F1
FOLLOWER+	82.2	75.1	78.5	67.8	68.7	68.2
MENTION+	**82.5**	**76.0**	**79.1**	67.5	**69.3**	68.4
RETWEET+	81.9	75.6	78.6	**69.0**	69.0	**69.0**

improvements over 400 but requires longer training time. Thus, we choose 400 as the size of word embeddings.

As presented in Table 5.4, NTEL-nonstruct performs 2.7% F1 worse than the NTEL baseline on the two test sets, which indicates the non-overlapping inference improves system performance on the task. With structured inference but without embeddings, NTEL performs roughly the same as S-MART, showing that a feedforward neural network offers similar expressivity to the regression trees employed by Yang and Chang [139].

Performance improves substantially with the incorporation of low-dimensional author, mention, and entity representations. As shown in Table 5.4, by learning the interactions between mention and entity representations, NTEL with mention-entity bilinear function outperforms the NTEL baseline system by 1.8% F1 on average. Specifically, the bilinear function results in considerable performance gains in recalls, with small compromise in precisions on the datasets.

Social information helps to increase about 1% F1 on top of both the NTEL baseline system and the NTEL system with mention-entity bilinear composition. In contrast to the mention-entity composition model, which mainly focuses on improving the baseline system on recall scores, the user-entity composition model increases around 2.5% recalls, without much sacrifice in precisions.

Our best system achieves the state-of-the-art results on the NEEL-test dataset and the TACL dataset, outperforming S-MART by 0.9% and 5.4% F1 scores respectively. To establish the statistical significance of the results, we obtain 100 bootstrap samples for each test set, and compute the F1 score on each sample for each algorithm. Two-tail paired t-test is then applied to determine if the F1 scores of two algorithms are significantly different. NTEL significantly outperforms S-MART on the NEEL-test dataset and the TACL dataset under $p < 0.01$ level, with t-statistics equal to 11.5 and 33.6 respectively.

As shown in Table 5.5, MENTION+ and RETWEET+ perform slightly better than FOLLOWER+. Puniyani *et al.* [35] show that the mention network has stronger linguistic properties than the follower network, as it gives better correlations on each author's distribution over latent topics as induced by latent Dirichlet allocation [147]. Our results suggest that the properties hold with respect to the authors' interests on real world entities.

Error analysis & discussion

We examine the outputs of different systems, focusing on investigating what errors are corrected by the two bilinear functions. The results reveal that the mention-entity composition improves the system ability to tackle mentions that are abbreviations such as 'WSJ' (*The Wall Street Journal*) and 'SJSU' (*San Jose State University*), which leads to higher recall scores. The mention-entity model also helps to eliminate errors that incorrectly link non-entities to popular entities. For example, the NTEL baseline system links 'sec' in the tweet 'I'm a be in Miami for sec to hit da radio!' to *Southeastern Conference*, which is corrected by the mention-entity composition model. The word semantic information encoded

in the mention representations alleviates the biased entity information given by the surface features.

The user-entity composition model is good at handling highly ambiguous mentions. For example, our full model successfully disambiguates entities for mentions such as 'Sox' (*Boston Red Sox* vs. *Chicago White Sox*), 'Sanders' (*Bernie Sanders* vs. *Barry Sanders*), and 'Memphis' (*Memphis Grizzlies* vs. *Memphis, Tennessee*), which are mistakenly linked to the other entities or **Nil** by the mention-entity model. Another example is that the social network information helps the system correctly link 'Kim' to *Lil' Kim* instead of *Kim Kardashian*, despite that the latter entity's wikipedia page is considerably more popular.

5.2 Sentiment Analysis with Social Attention

The meanings of some words can vary across different communities, which arises particular difficulties for NLP tasks like sentiment analysis. Fortunately, these differences are rarely idiosyncratic, but are often linked to social factors, such as age [49], gender [50], race [51], geography [52], and more ineffable characteristics such as political and cultural attitudes [148, 149]. For example, Hovy [30] shows that the accuracy of sentiment analysis and topic classification can be improved by the inclusion of coarse-grained author demographics such as age and gender.

However, such demographic information is not directly available in most datasets, and it is not yet clear whether predicted age and gender offers any improvements. On the other end of the spectrum are attempts to create *personalized* language technologies, as are often employed in information retrieval [31], recommender systems [32], and language modeling [33]. But personalization requires annotated data for each individual user—something that may be possible in interactive settings such as information retrieval, but is not typically feasible in natural language processing.

In this section, we assume that sentiment meaning is relatively consistent within socially nearby authors—an assumption we call *linguistic homophily*. To scale this basic

Table 5.6: Statistics of the SemEval Twitter sentiment datasets.

Dataset	# Positive	# Negative	# Neutral	# Tweet
Train 2013	3,230	1,265	4,109	8,604
Dev 2013	477	273	614	1,364
Test 2013	1,572	601	1,640	3,813
Test 2014	982	202	669	1,853
Test 2015	1,038	365	987	2,390

intuition to datasets with tens of thousands of unique authors, we compress the social network into vector representations of each author node, using an embedding method for large scale networks [137]. We then incorporate these embeddings into an attention-based neural network model, called SOCIAL ATTENTION, which employs multiple basis models to focus on different regions of the social network. We apply SOCIAL ATTENTION to Twitter sentiment classification, gathering social network metadata for Twitter users in the SemEval Twitter sentiment analysis tasks [150]. We further adopt the system to Ciao product reviews [151], training author embeddings using trust relationships between reviewers. SOCIAL ATTENTION offers a 2-3% improvement over related neural and ensemble architectures in which the social information is ablated. It also outperforms all prior published results on the SemEval Twitter test sets.

5.2.1 Data

In the SemEval Twitter sentiment analysis tasks, the goal is to classify the sentiment of each message as positive, negative, or neutral. Following Rosenthal *et al.* [152], we train and tune our systems on the SemEval Twitter 2013 training and development datasets respectively, and evaluate on the 2013–2015 SemEval Twitter test sets. Statistics of these datasets are presented in Table 5.6. Our training and development datasets lack some of the original Twitter messages, which may have been deleted since the datasets were constructed. However, our test datasets contain all the tweets used in the SemEval evaluations, making our results comparable with prior work.

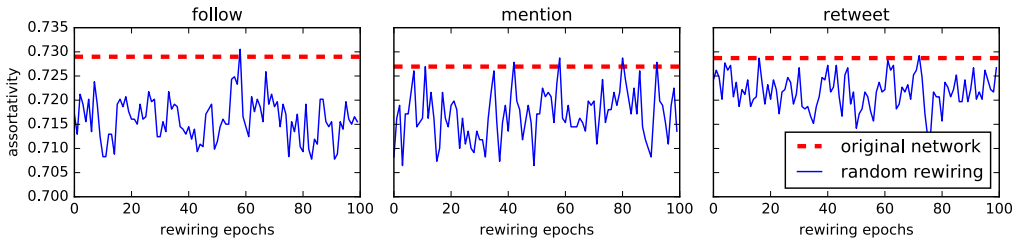


Figure 5.3: Assortativity of observed and randomized networks. Each rewiring epoch performs a number of rewiring operations equal to the total number of edges in the network. The randomly rewired networks almost always display lower assortativities than the original network, indicating that the accuracy of the lexicon-based sentiment analyzer is more assortative on the observed social network than one would expect by chance.

We construct three author social networks based on the follow, mention, and retweet relations between the 7,438 authors in the training dataset, which we refer as FOLLOWER, MENTION and RETWEET.[8] Specifically, we use the Twitter API to crawl the friends of the SemEval users (individuals that they follow) and the most recent 3,200 tweets in their timelines.[9] The mention and retweet links are then extracted from the tweet text and metadata. We treat all social networks as undirected graphs, where two users are socially connected if there exists at least one social relation between them.

5.2.2 Linguistic homophily

The hypothesis of *linguistic homophily* is that socially connected individuals tend to use language similarly, as compared to a randomly selected pair of individuals who are not socially connected. We now describe a pilot study that provides support for this hypothesis, focusing on the domain of sentiment analysis. The purpose of this study is to test whether errors in sentiment analysis are *assortative* on the social networks defined in the previous section: that is, if two individuals (i, j) are connected in the network, then a classifier error on i suggests that errors on j are more likely.

We test this idea using a simple lexicon-based classification approach, which we apply

[8]We could not gather the authorship information of 10% of the tweets in the training data, because the tweets or user accounts had been deleted by the time we crawled the social information.

[9]The Twitter API returns a maximum of 3,200 tweets.

to the SemEval training data, focusing only on messages that are labeled as positive or negative (ignoring the neutral class), and excluding authors who contributed more than one message (a tiny minority). Using the social media sentiment lexicons defined by Tang *et al.* [153],[10] we label a message as positive if it has at least as many positive words as negative words, and as negative otherwise.[11] The assortativity is the fraction of dyads for which the classifier makes two correct predictions or two incorrect predictions [154]. This measures whether classification errors are clustered on the network.

We compare the observed assortativity against the assortativity in a network that has been randomly rewired.[12] Each rewiring epoch involves a number of random rewiring operations equal to the total number of edges in the network. (The edges are randomly selected, so a given edge may not be rewired in each epoch.) By counting the number of edges that occur in both the original and rewired networks, we observe that this process converges to a steady state after three or four epochs. As shown in Figure 5.3, the original observed network displays more assortativity than the randomly rewired networks in nearly every case. Thus, the Twitter social networks display more linguistic homophily than we would expect due to chance alone.

The differences in assortativity across network types are small, indicating that none of the networks are clearly best. The retweet network was the most difficult to rewire, with the greatest proportion of shared edges between the original and rewired networks. This may explain why the assortativities of the randomly rewired networks were closest to the observed network in this case.

[10]The lexicons include words that are assigned at least 0.99 confidence by the method of Tang *et al.* [153]: 1,474 positive and 1,956 negative words in total.

[11]Ties go to the positive class because it is more common.

[12]Specifically, we use the `double_edge_swap` operation of the `networkx` package [155]. This operation preserves the degree of each node in the network.

5.2.3 Model

In this section, we describe a neural network method that leverages social network information to improve text classification. Our approach is inspired by ensemble learning, where the system prediction is the weighted combination of the outputs of several basis models. We encourage each basis model to focus on a local region of the social network, so that classification on socially connected individuals employs similar model combinations.

Given a set of instances $\{\mathbf{x}_i\}$ and authors $\{a_i\}$, the goal of personalized probabilistic classification is to estimate a conditional label distribution $p(y \mid \mathbf{x}, a)$. For most authors, no labeled data is available, so it is impossible to estimate this distribution directly. We therefore make a smoothness assumption over a social network G: individuals who are socially proximate in G should have similar classifiers. This idea is put into practice by modeling the conditional label distribution as a mixture over the predictions of K basis classifiers,

$$p(y \mid \mathbf{x}, a) = \sum_{k=1}^{K} \Pr(Z_a = k \mid a, G) \times p(y \mid \mathbf{x}, Z_a = k). \tag{5.13}$$

The basis classifiers $p(y \mid \mathbf{x}, Z_a = k)$ can be arbitrary conditional distributions; we use convolutional neural networks, as described in subsubsection 5.2.3. The component weighting distribution $\Pr(Z_a = k \mid a, G)$ is conditioned on the social network G, and functions as an attentional mechanism, described in subsubsection 5.2.3. The basic intuition is that for a pair of authors a_i and a_j who are nearby in the social network G, the prediction rules should behave similarly if the attentional distributions are similar, $p(z \mid a_i, G) \approx p(z \mid a_j, G)$. If we have labeled data only for a_i, some of the personalization from that data will be shared by a_j. The overall classification approach can be viewed as a mixture of experts [156], leveraging the social network as side information to choose the distribution over experts for each author.

Social attention model

The goal of the social attention model is to assign similar basis weights to authors who are nearby in the social network G. We operationalize social proximity by embedding each node's social network position into a vector representation. Specifically, we employ the LINE method [137], which estimates $D^{(v)}$ dimensional node embeddings $\mathbf{v}_a$ as parameters in a probabilistic model over edges in the social network. These embeddings are learned solely from the social network G, without leveraging any textual information. The attentional weights are then computed from the embeddings using a softmax layer,

$$\Pr(Z_a = k \mid a, G) = \frac{\exp\left(\phi_k^\top \mathbf{v}_a + b_k\right)}{\sum_{k'}^{K} \exp\left(\phi_{k'}^\top \mathbf{v}_a + b_{k'}\right)}. \tag{5.14}$$

This embedding method uses only single-relational networks; in the evaluation, we will show results for Twitter networks built from networks of follow, mention, and retweet relations. In future work, we may consider combining all of these relation types into a unified multi-relational network. It is possible that embeddings in such a network could be estimated using techniques borrowed from multi-relational knowledge networks [157, 158].

Sentiment classification with convolutional neural networks

We next describe the basis models, $p(y \mid \mathbf{x}, Z = k)$. Because our target task is classification on microtext documents, we model this distribution using convolutional neural networks (CNNs; Lecun *et al.* [159]), which have been proven to perform well on sentence classification tasks [160, 161]. CNNs apply layers of convolving filters to n-grams, thereby generating a vector of dense local features. CNNs improve upon traditional bag-of-words models because of their ability to capture word ordering information.

Let $\mathbf{x} = [\mathbf{h}_1, \mathbf{h}_2, \cdots, \mathbf{h}_n]$ be the input sentence, where $\mathbf{h}_i$ is the $D^{(w)}$ dimensional word vector corresponding to the i-th word in the sentence. We use one convolutional layer and

one max pooling layer to generate the sentence representation of $\mathbf{x}$. The convolutional layer involves filters that are applied to bigrams to produce feature maps. Formally, given the bigram word vectors $\mathbf{h}_i, \mathbf{h}_{i+1}$, the features generated by m filters can be computed by

$$\mathbf{c}_i = \tanh(\mathbf{W}_L \mathbf{h}_i + \mathbf{W}_R \mathbf{h}_{i+1} + \mathbf{b}), \tag{5.15}$$

where $\mathbf{c}_i$ is an m dimensional vector, $\mathbf{W}_L$ and $\mathbf{W}_R$ are $m \times D^{(w)}$ projection matrices, and $\mathbf{b}$ is the bias vector. The m dimensional vector representation of the sentence is given by the pooling operation

$$\mathbf{s} = \max_{i \in 1, \cdots, n-1} \mathbf{c}_i. \tag{5.16}$$

To obtain the conditional label probability, we utilize a multiclass logistic regression model,

$$\Pr(Y = t \mid \mathbf{x}, Z = k) = \frac{\exp(\boldsymbol{\beta}_t^\top \mathbf{s}_k + \beta_t)}{\sum_{t'=1}^{T} \exp(\boldsymbol{\beta}_{t'}^\top \mathbf{s}_k + \beta_{t'})}, \tag{5.17}$$

where $\boldsymbol{\beta}_t$ is an m dimensional weight vector, β_t is the corresponding bias term, and $\mathbf{s}_k$ is the m dimensional sentence representation produced by the k-th basis model.

Training

We fix the pretrained author and word embeddings during training our social attention model. Let Θ denote the parameters that need to be learned, which include $\{\mathbf{W}_L, \mathbf{W}_R, \mathbf{b}, \{\boldsymbol{\beta}_t, \beta_t\}_{t=1}^T\}$ for every basis CNN model, and the attentional weights $\{\phi_k, b_k\}_{k=1}^K$. We minimize the following logistic loss objective for each training instance:

$$\ell(\Theta) = -\sum_{t=1}^{T} \mathbf{1}[Y^* = t] \log \Pr(Y = t \mid \mathbf{x}, a), \tag{5.18}$$

where Y^* is the ground truth class for $\mathbf{x}$, and $\mathbf{1}[\cdot]$ represents an indicator function. We train the models for between 10 and 15 epochs using the Adam optimizer [162], with early stopping on the development set.

Initialization

One potential problem is that after initialization, a small number of basis models may claim most of the mixture weights for all the users, while other basis models are inactive. This can occur because some basis models may be initialized with parameters that are globally superior. As a result, the "dead" basis models will receive near-zero gradient updates, and therefore can never improve. The true model capacity can thereby be substantially lower than the K assigned experts.

Ideally, dead basis models will be avoided because each basis model should focus on a unique region of the social network. To ensure that this happens, we pretrain the basis models using an instance weighting approach from the domain adaptation literature [163]. For each basis model k, each author a has an instance weight $\alpha_{a,k}$. These instance weights are based on the author's social network node embedding, so that socially proximate authors will have high weights for the same basis models. This is ensured by endowing each basis model with a random vector $\boldsymbol{\gamma}_k \sim N(\mathbf{0}, \sigma^2 \mathbb{I})$, and setting the instance weights as,

$$\alpha_{a,k} = \text{sigmoid}(\boldsymbol{\gamma}_k^\top \mathbf{v}_a). \tag{5.19}$$

The simple design results in similar instance weights for socially proximate authors. During pre-training, we train the k-th basis model by optimizing the following loss function for every instance:

$$\ell_k = -\alpha_{a,k} \sum_{t=1}^{T} \mathbf{1}[Y^* = t] \log \Pr(Y = t \mid \mathbf{x}, Z_a = k). \tag{5.20}$$

The pretrained basis models are then assembled together and jointly trained using Equation 5.18.

Table 5.7: Statistics of the author social networks used for training author embeddings.

Network	# Author	# Relation
FOLLOWER+	18,281	1,287,260
MENTION+	25,007	1,403,369
RETWEET+	35,376	2,194,319

5.2.4 Experiments

Our main evaluation focuses on the 2013–2015 SemEval Twitter sentiment analysis tasks. The datasets have been described in subsection 5.2.1. We train and tune our systems on the Train 2013 and Dev 2013 datasets respectively, and evaluate on the Test 2013–2015 sets. In addition, we evaluate on another dataset based on Ciao product reviews [151].

Social network expansion

We utilize Twitter's follower, mention, and retweet social networks to train user embeddings. By querying the Twitter API in April 2015, we were able to identify 15,221 authors for the tweets in the SemEval datasets described above. We induce social networks for these individuals by crawling their friend links and timelines, as described in subsection 5.2.1. Unfortunately, these networks are relatively sparse, with a large amount of isolated author nodes. To improve the quality of the author embeddings, we expand the set of author nodes by adding nodes that do the most to densify the author networks: for the follower network, we add additional individuals that are followed by at least a hundred authors in the original set; for the mention and retweet networks, we add all users that have been mentioned or retweeted by at least twenty authors in the original set. The statistics of the resulting networks are presented in Table 5.7.

Experimental settings

We employ the pretrained word embeddings used by Astudillo *et al.* [164], which are trained with a corpus of 52 million tweets, and have been shown to perform very well

on this task. The embeddings are learned using the structured skip-gram model [125], and the embedding dimension is set as 600, following Astudillo *et al.* [164]. We report the same evaluation metric as the SemEval challenge: the average F1 score of positive and negative classes.[13]

Competitive systems We consider five competitive Twitter sentiment classification methods. *Convolutional neural network* (CNN) has been described in subsubsection 5.2.3, and is the basis model of SOCIAL ATTENTION. *Mixture of experts* employs the same CNN model as an expert, but the mixture densities solely depend on the input values. We adopt the summation of the pretrained word embeddings as the sentence-level input to learn the gating function.[14] The model architecture of *random attention* is nearly identical to SOCIAL ATTENTION: the only distinction is that we replace the pretrained author embeddings with random embedding vectors, drawing uniformly from the interval $(-0.25, 0.25)$. *Concatenation* concatenates the author embedding with the sentence representation obtained from CNN, and then feeds the new representation to a softmax classifier. Finally, we include SOCIAL ATTENTION, the attention-based neural network method described in subsection 5.2.3.

We also compare against the three top-performing systems in the SemEval 2015 Twitter sentiment analysis challenge [152]: WEBIS [165], UNITN [166], and LSISLIF [167]. UNITN achieves the best average F1 score on Test 2013–2015 sets among all the submitted systems. Finally, we republish results of NLSE [164], a non-linear subspace embedding model.

Parameter tuning We tune all the hyperparameters on the SemEval 2013 development set. We choose the number of bigram filters for the CNN models from {50, 100, 150}. The size of author embeddings is selected from {50, 100}. For *mixture of experts*, *random*

[13]Regarding the neutral class: systems are penalized with false positives when neutral tweets are incorrectly classified as positive or negative, and with false negatives when positive or negative tweets are incorrectly classified as neutral. This follows the evaluation procedure of the SemEval challenge.

[14]The summation of the pretrained word embeddings works better than the average of the word embeddings.

attention and SOCIAL ATTENTION, we compare a range of numbers of basis models, {3, 5, 10, 15}. We found that a relatively small number of basis models are usually sufficient to achieve good performance. The number of pretraining epochs is selected from {1, 2, 3}. During joint training, we check the performance on the development set after each epoch to perform early stopping.

Table 5.8: Average F1 score on the SemEval test sets. The best results are in **bold**. Results are marked with * if they are significantly better than CNN at $p < 0.05$.

System	Test 2013	Test 2014	Test 2015	Average
Our implementation				
CNN	69.31	72.73	63.24	68.43
Mixture of experts	68.97	72.07	64.28*	68.44
Random attention	69.48	71.56	64.37*	68.47
SOCIAL ATTENTION	71.91*	**75.07***	**66.75***	**71.24**
Reported results				
NLSE	72.09	73.64	65.21	70.31
WEBIS	68.49	70.86	64.84	68.06
UNITN	**72.79**	73.60	64.59	70.33
LSISLIF	71.34	71.54	64.27	69.05

Table 5.9: Comparison of different social networks with SOCIAL ATTENTION. The best results are in **bold**.

Network	Test 2013	Test 2014	Test 2015	Average
FOLLOWER+	71.49	74.17	66.00	70.55
MENTION+	71.72	74.14	66.27	70.71
RETWEET+	**71.91**	**75.07**	**66.75**	**71.24**

Results

Table 5.8 summarizes the main empirical findings, where we report results obtained from author embeddings trained on RETWEET+ network for SOCIAL ATTENTION. The results of different social networks for SOCIAL ATTENTION are shown in Table 5.9. The best hyperparameters are: 100 bigram filters; 100-dimensional author embeddings; $K = 5$ basis models; 1 pre-training epoch. To establish the statistical significance of the results,

Table 5.10: Average F1 score on the HDeg and the LDeg test sets.

System	HDeg	LDeg
CNN	64.40	69.83
SOCIAL ATTENTION	68.15	72.47

we obtain 100 bootstrap samples for each test set, and compute the F1 score on each sample for each algorithm. A two-tail paired t-test is then applied to determine if the F1 scores of two algorithms are significantly different, $p < 0.05$.

Mixture of experts, random attention, and CNN all achieve similar average F1 scores on the SemEval Twitter 2013–2015 test sets. Note that random attention can benefit from some of the personalized information encoded in the random author embeddings, as Twitter messages posted by the same author share the same attentional weights. However, it barely improves the results, because the majority of authors contribute a single message in the SemEval datasets. With the incorporation of author social network information, concatenation slightly improves the classification performance. Finally, SOCIAL ATTENTION gives much better results than concatenation, as it is able to model the interactions between text representations and author representations. It significantly outperforms CNN on all the SemEval test sets, yielding 2.8% improvement on average F1 score. SOCIAL ATTENTION also performs substantially better than the top-performing SemEval systems and NLSE, especially on the 2014 and 2015 test sets.

We now turn to a comparison of the social networks. As shown in Table 5.9, the RETWEET+ network is the most effective, although the differences are small: SOCIAL ATTENTION outperforms prior work regardless of which network is selected. Twitter's "following" relation is a relatively low-cost form of social engagement, and it is less public than retweeting or mentioning another user. Thus it is unsurprising that the follower network is least useful for socially-informed personalization. The RETWEET+ network has denser social connections than MENTION+, which could lead to better author embeddings.

Table 5.11: Top 5 more positive/negative words for the basis models in the SemEval training data. **Bolded** entries correspond to words that are often used ironically, by top authors related to basis model 1 and 4. Underlined entries are swear words, which are sometimes used positively by top users corresponding to basis model 3. *Italic* entries refer to celebrities and their fans, which usually appear in negative tweets by top authors for basis model 5.

Basis model	More positive	More negative
1	banging loss fever broken fucking	**dear like** god **yeah wow**
2	chilling cold ill sick suck	satisfy trust wealth strong **lmao**
3	ass damn piss bitch shit	talent honestly voting win clever
4	insane bawling fever weird cry	**lmao** super **lol haha hahaha**
5	ruin silly bad boring dreadful	*lovatics* wish *beliebers arianators kendall*

Error analysis For error analysis, we evenly split tweets in the SemEval Twitter 2013–2015 test sets into two new test sets based on the network degrees of the authors: the test set of the tweets posted by authors with higher network degrees is referred as HDeg; the test set of the tweets written by authors with lower network degrees is named as LDeg. The evaluation results of CNN and SOCIAL ATTENTION on HDeg and LDeg are presented in Table 5.10. As shown, both systems perform better on tweets of authors with fewer social relations. SOCIAL ATTENTION significantly improve average F1 scores on HDeg and LDeg, especially on the tweets by users with higher network degrees, as more social relations may lead to more accurate author embeddings.

Analysis

We now investigate whether language variation in sentiment meaning has been captured by different basis models. We focus on the same sentiment words [153] that we used to test linguistic homophily in our analysis. We are interested to discover sentiment words that are used with the opposite sentiment meanings by some authors. To measure the level of model-specificity for each word w, we compute the difference between the model-specific probabilities $p(y \mid X = w, Z = k)$ and the average probabilities of all basis models $\frac{1}{K}\sum_{k=1}^{K} p(y \mid X = w, Z = k)$ for positive and negative classes. The five words in the negative and positive lexicons with the highest scores for each model are presented in Ta-

Table 5.12: Tweet examples that contain sentiment words conveying specific sentiment meanings that differ from their common senses in the SemEval training data. The sentiment labels are adopted from the SemEval annotations.

Word	Sentiment	Example
fucking	positive	Aug 27th: UFC Rio..Anderson Silva vs Yushin Okami(last man to beat him)..Expect to see the best fighter on the planet put on a fucking show!
ill	positive	This time tomorrow night ill be partying it up in chapel hill :) #cantwait
sick	positive	Watch ESPN tonight to see me burning @user for a sick goal on the top ten. #realbackyardFIFA
bitch	positive	@user bitch u shoulda came with me Saturday sooooo much fun. Met Romeo santos lmao na i met his look a like
shit	positive	@user well shit! I hope your back for the morning show. I need you on my drive to Cupertino on Monday! Have fun!
dear	negative	Dear Spurs, You are out of COC, not in Champions League and come May wont be in top 4. Why do you even exist?
wow	negative	Wow. Tiger fires a 63 but not good enough. Nick Watney shoots a 59 if he birdies the 18th?!? #sick
lmao	negative	I just realized that Thalia is going to be tortured tomorrow for an hour lmao
lol	negative	Lol super awkward if its hella foggy at Rim tomorrow and the games suppose to be on tv lol Uhhhh.. Where's the ball? Lol
haha	negative	@user haha was it as bad as your night in Barcelona with the bucket o chund!?

ble 5.11.

As shown in the table, Twitter users corresponding to basis models 1 and 4 often use some words ironically in their tweets. Basis model 3 tends to assign positive sentiment polarity to swear words, and Twitter users related to basis model 5 seem to be less fond of fans of certain celebrities. Finally, basis model 2 identifies Twitter users that we have described in the introduction—they often adopt general negative words like 'ill', 'sick', and 'suck' positively. Examples containing some of these words are shown in Table 5.12.

Sentiment analysis of product reviews

The labeled datasets for Twitter sentiment analysis are relatively small; to evaluate our method on a larger dataset, we utilize a product review dataset by Tang *et al.* [151]. The dataset consists of 257,682 reviews written by 10,569 users crawled from a popular product

Table 5.13: Statistics of the Ciao product review datasets.

Dataset	# Author	# Positive	# Negative	# Review
Train Ciao	8,545	63,047	6,953	70,000
Dev Ciao	4,087	9,052	948	10,000
Test Ciao	5,740	17,978	2,022	20,000
Total	9,267	90,077	9,923	100,000

Table 5.14: Average F1 score on the Ciao test set. The best results are in **bold**. Results are marked with * and ** if they are significantly better than CNN and random attention respectively, at $p < 0.05$.

System	Test Ciao
CNN	78.43
Mixture of experts	78.37
Random attention	79.43*
Concatenation	77.99
SOCIAL ATTENTION	**80.19****

review sites, Ciao.[15] The rating information in discrete five-star range is available for the reviews, which is treated as the ground truth label information for the reviews. Moreover, the users of this site can mark explicit "trust" relationships with each other, creating a social network.

To select examples from this dataset, we first removed reviews that were marked by readers as "not useful." We treated reviews with more than three stars as positive, and less than three stars as negative; reviews with exactly three stars were removed. We then sampled 100,000 reviews from this set, and split them randomly into training (70%), development (10%) and test sets (20%). The statistics of the resulting datasets are presented in Table 5.13. We utilize 145,828 trust relations between 18,999 Ciao users to train the author embeddings. We consider the 10,000 most frequent words in the datasets, and assign them pretrained word2vec embeddings.[16] As shown in Table 5.13, the datasets have highly skewed class distributions. Thus, we use the average F1 score of positive and negative classes as the evaluation metic.

[15] `http://www.ciao.co.uk`
[16] `https://code.google.com/archive/p/word2vec`

The evaluation results are presented in Table 5.14. The best hyperparameters are generally the same as those for Twitter sentiment analysis, except that the optimal number of basis models is 10, and the optimal number of pretraining epochs is 2. Mixture of experts and concatenation obtain slightly worse F1 scores than the baseline CNN system, but random attention performs significantly better. In contrast to the SemEval datasets, individual users often contribute multiple reviews in the Ciao datasets (the average number of reviews from an author is 10.8; Table 5.13). As an author tends to express similar opinions toward related products, random attention is able to leverage the personalized information to improve sentiment analysis. Prior work has investigated the direction, obtaining positive results using speaker adaptation techniques [168]. Finally, by exploiting the social network of trust relations, SOCIAL ATTENTION obtains further improvements, outperforming random attention by a small but significant margin.

5.3 Related Work

Tweet entity linking Previous work on entity linking mainly focuses on well-written documents [169, 170, 171], where entity disambiguation is usually performed by maximizing the global topical coherence between entities. However, these approaches often yield unsatisfactory performance on Twitter messages, due to the short and noisy nature of the tweets. To tackle this problem, collective tweet entity linking methods that leverage enriched context and metadata information have been proposed [172]. Guo *et al.* [173] search for textually similar tweets for a target tweet, and encourage these Twitter messages to contain similar entities through label propagation. Shen *et al.* [174] employ Twitter user account information to improve entity linking, based on the intuition that all tweets posted by the same user share an underlying topic distribution. Fang and Chang [141] demonstrate that spatial and temporal signals are critical for the task, and they advance the performance by associating entity prior distributions with different timestamps and locations. Our work overcomes the difficulty by leveraging social relations—socially connected individuals are

assumed to share similar interests on entities. As the Twitter post information is often sparse for some users, our assumption enables the utilization of more relevant information that helps to improve the task.

Structured learning models Many NLP tasks such as Part-of-Speech (POS) tagging, dependency parsing are inherently structured learning problems. Early research in NLP has been focused on linear structured learning methods such as Structured Perceptron [175], Conditional Random Field [71], and linear Structured SVM [176, 177]. However, the linear models are not expressive enough to capture the non-linear relationships between dense features, which results in poor performance on the task of tweet entity linking [139]. Non-linear structured learning models have been successfully applied in many NLP tasks recently, including neural network models [178] and tree-based models [139]. We show a carefully designed neural structured learning model based on loss-augmented training gives competitive performance on tweet entity linking. Furthermore, our approach is flexible to leverage valuable metadata information such as social relations, which leads to state-of-the-art results on the task.

Sentiment analysis with social relations Previous work on incorporating social relations for sentiment classification mainly relies on the label consistency assumption, where the existence of social connections between users is considered as a clue that the sentiment polarities of all messages from the users should be similar. Speriosu *et al.* [179] construct a heterogeneous network with tweets, users, and n-grams as nodes, and each node is associated with a sentiment label distribution obtained from a maximum entropy classifier or sentiment lexicons. The label distributions of tweets is then refined by performing label propagation over social relations. Hu *et al.* [40] model social relations using the graph Laplacian of the adjacency graph representation of the social network, which they employ as a source of regularization, so that socially-similar users are encouraged to have similar labels. Tan *et al.* [39] leverage a similar intuition, using a factor graph based approach in

which the labels of targets belonging to socially connected users are treated as factors in a joint probabilistic model. Our work is based on a different intuition: rather than assuming that labels will tend to be similar for socially-connected users, we assume that similar usage of language. These assumptions are complementary; if both hold for a specific setting, then label consistency and linguistic consistency could in principle be applied to improve performance.

Ensemble learning Ensemble learning [180] is a popular machine learning method, which aggregates a set of individually trained models, and makes a combined prediction. Previous research has shown that an ensemble is often more accurate than any of the single models in the ensemble. Bootstrap aggregating (Bagging; Breiman, 1996) and Random Forest [182] are two popular ensemble methods designed to improve the stability and accuracy of machine learning algorithms by reducing variance that helps to avoid overfitting. Bagging learns basis models with bootstrap samples and makes the prediction based on techniques like major voting. Random Forest improves upon Bagging by only adopting a subsample of features to train each basis model. AdaBoost [183] adaptively trains a sequence of models, so that subsequent models are tweaked in favor of those instances misclassified by previous models. SOCIAL ATTENTION can be viewed as an ensemble method in the sense that it relies on different models to make a prediction. However, in contrast to standard ensemble approaches, SOCIAL ATTENTION leverages additional social relation information to guide individual model to sample labeled training data.

CHAPTER 6
CONCLUSION

The goal of this thesis is to deliver reliable and effective NLP tools to non-standard domains, such as social media and digital humanities. In order to overcome variation in non-standard languages, we present three complementary approaches that cope with different respects and levels of the variation. Chapter 3 focuses on tackling lexical variation with text normalization, which has been shown to improve many underlying NLP tasks. A novel method for unsupervised multi-domain adaptation is presented in chapter 4, which is able to address a full range of linguistic variation, including the lexicon, syntax, and semantics. Domain adaptation targets a specific NLP task, where we utilize the part-of-speech tagging task to demonstrate the effectiveness of our approach. For text normalization, we have presented a unified, unsupervised statistical model for normalizing social media text, attaining the best reported performance on the two standard normalization datasets. The power of our approach comes from flexible modeling of word-to-word relationships through features, while exploiting contextual regularity to train the corresponding feature. For domain adaptation, we have shown that a simple feature embedding approach can offer strong performance, avoid practical drawbacks of alternative representation learning approaches, and are easy to learn using existing word embedding methods.

Text normalization and domain adaptation are both well studied problems in previous work. However, they ignore a subtle challenge posed by language variation—different authors use language differently. We address individual-level linguistic variation in chapter 5, and improve microblog entity linking and sentiment analysis by leveraging social structure information. We explore the social theory of homophily, which entails that social neighbors are more likely to have similar behaviors or share similar interests. In order to adapt the theory to specific NLP tasks, we first train distributed author representations

using social relation information. Our microblog entity linking system employs the entity homophily assumption and models the compositions of vector representations of authors and entities. By exploiting the social network as a source of contextual information, the system achieves the state-of-the-art results on two benchmark datasets. We make the linguistic homophily assumption for the task of sentiment analysis. By learning basis models focusing on different local regions of the social network, our approach is able to capture subtle shifts in meaning for individual words across social network connections. Inspired by ensemble learning methods, we have formulated this model by employing social attention mechanism—the final prediction is the weighted combination of the outputs of the basis models, depending on the social network structure. Our model achieves significant improvements over standard convolutional networks and related ensemble architectures in which the social information is ablated.

For future work, we would like to combine different approaches to further improve NLP systems when adopted to non-standard texts. We have some successful experiences of combining text normalization and domain adaptation [43]. Since socially adapted NLP focuses on individual-level variation that is different from the other approaches, it makes sense to combine text normalization or domain adaptation with socially aware NLP techniques. One natural direction for future work is to apply socially adapted NLP methods from chapter 5 to social networks in other settings besides microblogs, such as webpages and academic research articles. We would also like to investigate other metadata attributes that are relevant to the tasks, such as spatial and temporal signals for entity linking and demographic factors for sentiment analysis. Another key question for future work is whether the task and social dimensions can be decoupled: can we learn a socially-infused model that is useful across multiple tasks? Answering this question requires ground truth annotations for multiple tasks for a set of socially linked authors.

Appendices

APPENDIX A

APPENDIX: TAG MAPPINGS

As discussed in chapter 4, this table provides the full mapping from Penn-Corpus of Historical English tags to Penn Treebank Tags used in our evaluation.

PCHE → PTB	PCHE → PTB	PCHE → PTB
, (sent-internal) → , (comma)	ELSE → RB	OTHER → PRP
. (sent-final) → . (sent-final)	EX → EX	OTHER$ → PRP
' (single quote) → '' (closing quote)	FOR → IN	OTHERS$ → PRP
¨ (double quote) → '' (closing quote)	FP → CC	OTHERS → PRP
$ → PRP$	FW → FW	P → IN
ADJ → JJ	HAG → VBG	PRO → PRP
ADJR → JJR	HAN → VBN	PRO$ → PRP$
ADJS → JJS	HV → VB	Q → JJ
ADV → RB	HVD → VBD	QS → RBS
ADVR → RBR	HVI → VB	QR → RBR
ADVS → RBS	HVN → VBN	RP → RB
ALSO → RB	HVP → VBP	SUCH → RB
BAG → VBG	INTJ → UH	TO → TO
BE → VB	MD → MD	VAG → VBG
BED → VBD	N → NN	VAN → VBN
BEI → VB	N$ → NN	VB → VB
BEN → VBN	NEG → RB	VBD → VBD
BEP → VBZ	NPR → NNP	VBI → VB
C → IN	NPR$ → NNP	VBN → VBN
CONJ → CC	NPRS → NNPS	VBP → VBP
D → DT	NPRS$ → NNPS	WADV → WRB
DAG → VBG	NS → NNS	WARD → VB
DAN → VBN	NS$ → NNS	WD → WDT
DO → VB	NUM → CD	WPRO → WP
DOD → VBD	NUM$ → CD	WPRO$ → WP$
DOI → VB	ONE → PRP	WQ → IN
DON → VBN	ONES → PRP	X → X
DOP → VBP	ONE$ → PRP$	

APPENDIX B

APPENDIX: ADDITIONAL ENTITY LINKING RESULTS

Table B.1: Evaluation results on the NEEL-test and TACL datasets for different systems. Twitter messages that contain no ground truth entities are excluded for both training and testing. The best results are in **bold**.

System	user -entity	mention -entity	NEEL-test			TACL			Avg. F1
			P	R	F1	P	R	F1	
Our approach									
NTEL-nonstruct			83.0	71.8	77.0	80.9	69.0	74.5	75.8
NTEL			84.4	73.9	78.8	82.0	71.3	76.3	77.6
NTEL	✓		83.8	76.7	80.1	81.8	**73.3**	77.3	78.7
NTEL		✓	84.1	78.3	81.1	83.0	71.7	76.9	79.0
NTEL	✓	✓	**84.8**	**79.3**	**82.0**	**83.5**	72.7	**77.7**	**79.9**
Best published results									
S-MART			83.2	79.2	81.1	76.8	73.0	74.9	78.0

This chapter provides additional results to support Chapter 5. In the first version of Yang and Chang [139], the Twitter messages that contain no ground truth entities are excluded in the experiments. For completeness, we now present the evaluation results of NTEL in this setting, which are shown in Table B.1. The RETWEET+ network is adopted to train author embeddings. The best hyper-parameters are the same as those described in subsection 5.1.4, except for the L2 regularization penalty for the composition parameters, which is set as 0.01 here.

The results are generally better than those presented in Table 5.4. As shown, NTEL benefits from the distributed representations of authors, mentions, and entities, which improve the average F1 score by 2.3 points. NTEL also gives the best results on the datasets, outperforming S-MART by about 2% F1 on average.

REFERENCES

[1] J.-B. Michel, Y. K. Shen, A. P. Aiden, A. Veres, M. K. Gray, J. P. Pickett, D. Hoiberg, D. Clancy, P. Norvig, J. Orwant, *et al.*, "Quantitative analysis of culture using millions of digitized books," *Science*, vol. 331, no. 6014, pp. 176–182, 2011.

[2] J. Eisenstein, B. O'Connor, N. A. Smith, and E. P. Xing, "A latent variable model for geographic lexical variation," in *Proceedings of Empirical Methods for Natural Language Processing (EMNLP)*, Cambridge, MA, 2010, pp. 1277–1287.

[3] S. Gouws, D. Metzler, C. Cai, and E. Hovy, "Contextual bearing on linguistic variation in social media," in *Proceedings of the Workshop on Language in Social Media (LSM)*, 2011.

[4] R. Giusti, A Candido, M. Muniz, L. Cucatto, and S. Aluísio, "Automatic detection of spelling variation in historical corpus," in *Proceedings of the Corpus Linguistics Conference (CL)*, 2007.

[5] L. Borin and M. Forsberg, "Something old, something new: A computational morphological description of old swedish," in *Proceedings of the LREC 2008 workshop on language technology for cultural heritage data (LaTeCH 2008)*, 2008, pp. 9–16.

[6] J. Eisenstein, "Systematic patterning in phonologically-motivated orthographic variation," *Journal of Sociolinguistics*, vol. 19, pp. 161–188, 2 2015.

[7] A. Johannsen, D. Hovy, and A. Søgaard, "Cross-lingual syntactic variation over age and gender," in *Proceedings of the Conference on Natural Language Learning (CoNLL)*, 2015.

[8] E. Eumeridou, B. Nkwenti-Azeh, and J. McNaught, "An analysis of verb subcategorization frames in three special language corpora with a view towards automatic term recognition," *Computers and the Humanities*, vol. 38, no. 1, pp. 37–60, 2004.

[9] R. Garside and N. Smith, "A hybrid grammatical tagger: Claws4," *Corpus annotation: Linguistic information from computer text corpora*, pp. 102–121, 1997.

[10] P. Rayson, D. Archer, A. Baron, J. Culpeper, and N. Smith, "Tagging the bard: Evaluating the accuracy of a modern pos tagger on early modern english corpora," in *Corpus Linguistics Conference*, 2007.

[11] J. R. Finkel, T. Grenager, and C. Manning, "Incorporating non-local information into information extraction systems by gibbs sampling," in *Proceedings of the As-*

sociation for Computational Linguistics (ACL), Ann Arbor, MI, 2005, pp. 363–370.

[12] A. Ritter, S. Clark, Mausam, and O. Etzioni, "Named entity recognition in tweets: An experimental study," in *Proceedings of Empirical Methods for Natural Language Processing (EMNLP)*, Edinburgh, UK, 2011.

[13] M. J. Paul and M. Dredze, "You are what you tweet: Analyzing twitter for public health.," in *Proceedings of the International Conference on Web and Social Media (ICWSM)*, 2011, pp. 265–272.

[14] M. Dredze, M. J. Paul, S. Bergsma, and H. Tran, "Carmen: A Twitter geolocation system with applications to public health," in *AAAI Workshop on Expanding the Boundaries of Health Informatics Using Artificial Intelligence*, 2013, pp. 20–24.

[15] M. Conover, J. Ratkiewicz, M. Francisco, B. Gonçalves, F. Menczer, and A. Flammini, "Political polarization on twitter," in *Proceedings of the International Conference on Web and Social Media (ICWSM)*, 2011.

[16] M. Iyyer, P. Enns, J. Boyd-Graber, and P. Resnik, "Political ideology detection using recursive neural networks," in *Proceedings of the Association for Computational Linguistics (ACL)*, Baltimore, MD, 2014.

[17] S. Asur and B. A. Huberman, "Predicting the future with social media," in *Web Intelligence and Intelligent Agent Technology (WI-IAT)*, 2010, pp. 492–499.

[18] D. Bamman, T. Underwood, and N. A. Smith, "A bayesian mixed effects model of literary character," in *Proceedings of the Association for Computational Linguistics (ACL)*, Baltimore, MD, 2014, pp. 370–379.

[19] B. Han, P. Cook, and T. Baldwin, "Lexical normalization for social media text," *ACM Transactions on Intelligent Systems and Technology*, vol. 4, no. 1, p. 5, 2013.

[20] H. Hassan and A. Menezes, "Social text normalization using contextual graph random walks," in *Proceedings of the Association for Computational Linguistics (ACL)*, Sophia, Bulgaria, 2013.

[21] B. Huberman, D. M. Romero, and F. Wu, "Social networks that matter: Twitter under the microscope," *First Monday*, vol. 14, no. 1, 2008.

[22] M. Thomas, B. Pang, and L. Lee, "Get out the vote: Determining support or opposition from Congressional floor-debate transcripts," in *Proceedings of Empirical Methods for Natural Language Processing (EMNLP)*, 2006, pp. 327–335.

[23] C. Winterer, "Where is america in the republic of letters?" *Modern Intellectual History*, vol. 9, no. 03, pp. 597–623, 2012.

[24] B. Han and T. Baldwin, "Lexical normalisation of short text messages: Makn sens a# twitter," in *Proceedings of the Association for Computational Linguistics (ACL)*, Portland, OR, 2011.

[25] J. Eisenstein, "What to do about bad language on the internet," in *Proceedings of the North American Chapter of the Association for Computational Linguistics (NAACL)*, Atlanta, GA, 2013, pp. 359–369.

[26] F. Liu, F. Weng, B. Wang, and Y. Liu, "Insertion, deletion, or substitution?: Normalizing text messages without pre-categorization nor supervision," in *Proceedings of the Association for Computational Linguistics (ACL)*, Portland, OR, 2011, pp. 71–76.

[27] C. Zhang, T. Baldwin, H. Ho, B. Kimelfeld, and Y. Li, "Adaptive parser-centric text normalization.," in *Proceedings of the Association for Computational Linguistics (ACL)*, Sophia, Bulgaria, 2013.

[28] A. Baron and P. Rayson, "Vard2: A tool for dealing with spelling variation in historical corpora," in *Postgraduate conference in corpus linguistics*, 2008.

[29] C. Galves and P. Faria, *Tycho brahe parsed corpus of historical portuguese*, `http://www.tycho.iel.unicamp.br/~tycho/corpus/en/index.html`, 2010.

[30] D. Hovy, "Demographic factors improve classification performance," in *Proceedings of the Association for Computational Linguistics (ACL)*, Beijing, China, 2015, pp. 752–762.

[31] X. Shen, B. Tan, and C. Zhai, "Implicit user modeling for personalized search," in *Proceedings of the 14th ACM international conference on Information and knowledge management*, ACM, 2005, pp. 824–831.

[32] J. Basilico and T. Hofmann, "Unifying collaborative and content-based filtering," in *Proceedings of the twenty-first international conference on Machine learning*, ACM, 2004.

[33] M. Federico, "Bayesian estimation methods for n-gram language model adaptation," in *Spoken Language, 1996. ICSLP 96. Proceedings., Fourth International Conference on*, IEEE, vol. 1, 1996, pp. 240–243.

[34] M. McPherson, L. Smith-Lovin, and J. M. Cook, "Birds of a feather: Homophily in social networks," *Annual review of sociology*, pp. 415–444, 2001.

[35] K. Puniyani, J. Eisenstein, S. Cohen, and E. P. Xing, "Social links from latent topics in microblogs," in *Proceedings of NAACL Workshop on Social Media*, Los Angeles, 2010.

[36] J. Bryden, S. Funk, and V. Jansen, "Word usage mirrors community structure in the online social network twitter," *EPJ Data Science*, vol. 2, no. 1, 3, 2013.

[37] M. Thelwall, "Homophily in MySpace," *J. Am. Soc. Inf. Sci.*, vol. 60, no. 2, pp. 219–231, 2009.

[38] F. Al Zamal, W. Liu, and D. Ruths, "Homophily and latent attribute inference: Inferring latent attributes of Twitter users from neighbors," in *Proceedings of the International Conference on Web and Social Media (ICWSM)*, 2012, pp. 387–390.

[39] C. Tan, L. Lee, J. Tang, L. Jiang, M. Zhou, and P. Li, "User-level sentiment analysis incorporating social networks," in *Proceedings of Knowledge Discovery and Data Mining (KDD)*, 2011.

[40] X. Hu, L. Tang, J. Tang, and H. Liu, "Exploiting social relations for sentiment analysis in microblogging," in *Proceedings of the sixth ACM international conference on Web search and data mining (WSDM)*, 2013, pp. 537–546.

[41] Y. Yang and J. Eisenstein, "A log-linear model for unsupervised text normalization," in *Proceedings of Empirical Methods for Natural Language Processing (EMNLP)*, Seattle, WA, 2013.

[42] ——, "Unsupervised multi-domain adaptation with feature embeddings," in *Proceedings of the North American Chapter of the Association for Computational Linguistics (NAACL)*, Denver, CO, 2015.

[43] ——, "Part-of-speech tagging for historical english," in *Proceedings of the North American Chapter of the Association for Computational Linguistics (NAACL)*, San Diego, CA, 2016.

[44] Y. Yang, M.-W. Chang, and J. Eisenstein, "Toward socially-infused information extraction: Embedding authors, mentions, and entities," in *Proceedings of Empirical Methods for Natural Language Processing (EMNLP)*, Austin, TX, 2016.

[45] Y. Yang and J. Eisenstein, "Overcoming language variation in sentiment analysis with social attention," *Transactions of the Association of Computational Linguistics (TACL)*, 2017.

[46] J. L. Dillard, *Black English: Its history and usage in the United States*. 1973.

[47] D. Bamman, J. Eisenstein, and T. Schnoebelen, "Gender identity and lexical variation in social media," *Journal of Sociolinguistics*, vol. 18, no. 2, pp. 135–160, 2014.

[48] P. Analytics, "Twitter study–august 2009," *San Antonio, TX: Pear Analytics. Available at: Www. pearanalytics. com/blog/wp-content/uploads/2010/05/Twitter-Study-August-2009. pdf*, 2009.

[49] S. Rosenthal and K. McKeown, "Age prediction in blogs: A study of style, content, and online behavior in pre- and Post-Social media generations," in *Proceedings of the Association for Computational Linguistics (ACL)*, Portland, OR, 2011, pp. 763–772.

[50] P. Eckert and S. McConnell-Ginet, *Language and Gender*. New York: Cambridge University Press, 2003.

[51] L. J. Green, *African American English: A Linguistic Introduction*. Cambridge, U.K.: Cambridge University Press, Sep. 2002.

[52] P. Trudgill, "Linguistic change and diffusion: Description and explanation in sociolinguistic dialect geography," *Language in Society*, vol. 3, no. 2, pp. 215–246, 1974.

[53] T. Ali, D. Schramm, M. Sokolova, and D. Inkpen, "Can i hear you? sentiment analysis on medical forums.," in *Proceedings of the International Joint Conference on Natural Language Processing (IJCNLP)*, 2013, pp. 667–673.

[54] H. Mao, S. Counts, and J. Bollen, "Computational economic and finance gauges: Polls, search, and twitter," in *Meeting of the National Bureau of Economic Research-Behavioral Finance Meeting, Stanford, CT*, 2011.

[55] M. Nagarajan, K. Gomadam, A. P. Sheth, A. Ranabahu, R. Mutharaju, and A. Jadhav, "Spatio-temporal-thematic analysis of citizen sensor data: Challenges and experiences," in *Proceedings of the International Conference on Web Information Systems Engineering*, Springer, 2009, pp. 539–553.

[56] D. Shamma, L. Kennedy, and E. Churchill, "Tweetgeist: Can the twitter timeline reveal the structure of broadcast events," in *Proceedings of Computer-Supported Cooperative Work (CSCW)*, 2010.

[57] M. P. Marcus, M. A. Marcinkiewicz, and B. Santorini, "Building a large annotated corpus of English: The penn treebank," *Computational Linguistics*, vol. 19, no. 2, pp. 313–330, 1993.

[58] E. F. Tjong Kim Sang and F. De Meulder, "Introduction to the conll-2003 shared task: Language-independent named entity recognition," in *Proceedings of the sev-*

enth conference on Natural language learning at HLT-NAACL 2003-Volume 4, Association for Computational Linguistics, 2003, pp. 142–147.

[59] K. Gimpel, N. Schneider, B. O'Connor, D. Das, D. Mills, J. Eisenstein, M. Heilman, D. Yogatama, J. Flanigan, and N. A. Smith, "Part-of-speech tagging for Twitter: Annotation, features, and experiments," in *Proceedings of the Association for Computational Linguistics (ACL)*, Portland, OR, 2011, pp. 42–47.

[60] O. Owoputi, B. O'Connor, C. Dyer, K. Gimpel, N. Schneider, and N. A. Smith, "Improved part-of-speech tagging for online conversational text with word clusters," in *Proceedings of the North American Chapter of the Association for Computational Linguistics (NAACL)*, Atlanta, GA, 2013, pp. 380–390.

[61] L. Kong, N. Schneider, S. Swayamdipta, A. Bhatia, C. Dyer, and N. A. Smith, "A dependency parser for tweets," in *Proceedings of Empirical Methods for Natural Language Processing (EMNLP)*, 2014.

[62] D. Ramage, D. Hall, R. Nallapati, and C. D. Manning, "Labeled lda: A supervised topic model for credit attribution in multi-labeled corpora," in *Proceedings of the Association for Computational Linguistics (ACL)*, Suntec, Singapore, 2009.

[63] R. Sproat, A. Black, S. Chen, S. Kumar, M. Ostendorf, and C. Richards, "Normalization of non-standard words," *Computer Speech & Language*, vol. 15, no. 3, pp. 287–333, 2001.

[64] M. Choudhury, R. Saraf, V. Jain, A. Mukherjee, S. Sarkar, and A. Basu, "Investigation and modeling of the structure of texting language," *International Journal of Document Analysis and Recognition (IJDAR)*, vol. 10, no. 3-4, pp. 157–174, 2007.

[65] A. Aw, M. Zhang, J. Xiao, and J. Su, "A phrase-based statistical model for SMS text normalization," in *Proceedings of the International Conference on Computational Linguistics (COLING)*, 2006, pp. 33–40.

[66] C. Kobus, F. Yvon, and G. Damnati, "Normalizing sms: Are two metaphors better than one?" In *Proceedings of the International Conference on Computational Linguistics (COLING)*, Manchester, UK, 2008, pp. 441–448.

[67] R. Beaufort, S. Roekhaut, L.-A. Cougnon, and C. Fairon, "A hybrid rule/model-based finite-state framework for normalizing sms messages," in *Proceedings of the Association for Computational Linguistics (ACL)*, Uppsala, Sweden, 2010, pp. 770–779.

[68] P. Cook and S. Stevenson, "An unsupervised model for text message normalization," in *Proceedings of the Workshop on Computational Approaches to Linguistic Creativity*, 2009, pp. 71–78.

[69] D. Contractor, T. A. Faruquie, and L. V. Subramaniam, "Unsupervised cleansing of noisy text," in *Proceedings of the International Conference on Computational Linguistics (COLING)*, 2010, pp. 189–196.

[70] S. Gouws, D. Hovy, and D. Metzler, "Unsupervised mining of lexical variants from noisy text," in *Proceedings of Empirical Methods for Natural Language Processing (EMNLP)*, Edinburgh, UK, 2011.

[71] J. Lafferty, A. McCallum, and F. Pereira, "Conditional random fields: Probabilistic models for segmenting and labeling sequence data," in *Proceedings of the International Conference on Machine Learning (ICML)*, 2001.

[72] F. Liu, F. Weng, and X. Jiang, "A broad-coverage normalization system for social media language," in *Proceedings of the Association for Computational Linguistics (ACL)*, Jeju, Korea, 2012, pp. 1035–1044.

[73] H. Daumé III, "Bayesian multitask learning with latent hierarchies," in *Proceedings of Uncertainty in Artificial Intelligence (UAI)*, 2009, pp. 135–142.

[74] J. R. Finkel and C. Manning, "Hierarchical bayesian domain adaptation," in *Proceedings of the North American Chapter of the Association for Computational Linguistics (NAACL)*, Boulder, CO, 2009, pp. 602–610.

[75] S. Ben-David, J. Blitzer, K. Crammer, A. Kulesza, F. Pereira, and J. W. Vaughan, "A theory of learning from different domains," *Machine learning*, vol. 79, no. 1-2, pp. 151–175, 2010.

[76] R. K. Ando and T. Zhang, "A framework for learning predictive structures from multiple tasks and unlabeled data," *The Journal of Machine Learning Research*, vol. 6, pp. 1817–1853, 2005.

[77] J. Blitzer, R. McDonald, and F. Pereira, "Domain adaptation with structural correspondence learning," in *Proceedings of Empirical Methods for Natural Language Processing (EMNLP)*, 2006, pp. 120–128.

[78] M. Chen, Z. Xu, K. Weinberger, and F. Sha, "Marginalized denoising autoencoders for domain adaptation," in *Proceedings of the International Conference on Machine Learning (ICML)*, 2012.

[79] K. Ganchev, J. Graça, J. Gillenwater, and B. Taskar, "Posterior regularization for structured latent variable models," *The Journal of Machine Learning Research*, vol. 11, pp. 2001–2049, 2010.

[80] F. Huang and A. Yates, "Biased representation learning for domain adaptation," in *Proceedings of Empirical Methods for Natural Language Processing (EMNLP)*, 2012, pp. 1313–1323.

[81] K. Ganchev and D. Das, "Cross-lingual discriminative learning of sequence models with posterior regularization," in *Proceedings of Empirical Methods for Natural Language Processing (EMNLP)*, Seattle, WA, 2013, pp. 1996–2006.

[82] N. A. Smith and J. Eisner, "Contrastive estimation: Training log-linear models on unlabeled data," in *Proceedings of the Association for Computational Linguistics (ACL)*, Ann Arbor, MI, 2005, pp. 354–362.

[83] T. Berg-Kirkpatrick, A. Bouchard-Côté, J. DeNero, and D. Klein, "Painless unsupervised learning with features," in *Proceedings of the North American Chapter of the Association for Computational Linguistics (NAACL)*, Los Angeles, CA, 2010, pp. 582–590.

[84] O. Cappe, S. J. Godsill, and E. Moulines, "An overview of existing methods and recent advances in sequential monte carlo," *Proceedings of the IEEE*, vol. 95, no. 5, pp. 899–924, May 2007.

[85] S. J. Godsill, A. Doucet, and M. West, "Monte carlo smoothing for non-linear time series," pp. 156–168, 2004.

[86] A. Doucet, N. Gordon, and V. Krishnamurthy, "Particle filters for state estimation of jump markov linear systems," *Trans. Sig. Proc.*, vol. 49, no. 3, pp. 613–624, Mar. 2001.

[87] S. Petrović, M. Osborne, and V. Lavrenko, "The edinburgh twitter corpus," in *Proceedings of the NAACL HLT Workshop on Computational Linguistics in a World of Social Media*, Los Angeles, California, 2010, pp. 25–26.

[88] A. Stolcke, "SRILM - an extensible language modeling toolkit," in *Proceedings of ICSLP*, 2002, pp. 901–904.

[89] J. Langford, L. Li, and T. Zhang, "Sparse online learning via truncated gradient," *The Journal of Machine Learning Research*, vol. 10, pp. 777–801, 2009.

[90] D. D. Lee and H. S. Seung, "Algorithms for non-negative matrix factorization," in *Advances in Neural Information Processing Systems (NIPS)*, Vancouver, 2001, pp. 556–562.

[91] S. Brody and N. Diakopoulos, "Cooooooooooooooollllllllllllll!!!!!!!!!!!!!!: Using word lengthening to detect sentiment in microblogs," in *Proceedings of Empiri-*

cal Methods for Natural Language Processing (EMNLP), Edinburgh, UK, 2011, pp. 562–570.

[92] J. Eisenstein, "Phonological factors in social media writing," in *Proceedings of the Workshop on Language Analysis in Social Media*, Atlanta, 2013, pp. 11–19.

[93] S. Tagliamonte and R. Temple, "New perspectives on an ol' variable: (t,d) in british english," *Language Variation and Change*, vol. 17, pp. 281–302, Sep. 2005.

[94] S. Argamon, M. Koppel, J. W. Pennebaker, and J. Schler, "Mining the blogosphere: Age, gender, and the varieties of self-expression," *First Monday*, vol. 12, no. 9, 2007.

[95] J. Eisenstein, N. A. Smith, and E. P. Xing, "Discovering sociolinguistic associations with structured sparsity," in *Proceedings of the Association for Computational Linguistics (ACL)*, Portland, OR, 2011, pp. 1365–1374.

[96] J. D. Burger, J. Henderson, G. Kim, and G. Zarrella, "Discriminating gender on twitter," in *Proceedings of Empirical Methods for Natural Language Processing (EMNLP)*, Edinburgh, UK, 2011.

[97] X. Glorot, A. Bordes, and Y. Bengio, "Domain adaptation for large-scale sentiment classification: A deep learning approach," in *Proceedings of the International Conference on Machine Learning (ICML)*, Seattle, WA, 2011.

[98] Y. Mansour, M. Mohri, and A. Rostamizadeh, "Domain adaptation with multiple sources," in *Advances in Neural Information Processing Systems (NIPS)*, 2009, pp. 1041–1048.

[99] A. Kroch, B. Santorini, and A. Diertani, *Penn-helsinki parsed corpus of early modern english*, `http://www.ling.upenn.edu/hist-corpora/PPCEME-RELEASE-2/index.html`, 2004.

[100] ——, *The penn parsed corpus of modern british english*, `http://www.ling.upenn.edu/hist-corpora/PPCMBE-RELEASE-1/index.html`, 2010.

[101] J. Blitzer, M. Dredze, and F. Pereira, "Biographies, bollywood, boom-boxes and blenders: Domain adaptation for sentiment classification," in *Proceedings of the Association for Computational Linguistics (ACL)*, Prague, 2007, pp. 440–447.

[102] N. A. Smith, "Linguistic structure prediction," *Synthesis Lectures on Human Language Technologies*, vol. 4, no. 2, pp. 1–274, 2011.

[103] J. Turian, L. Ratinov, and Y. Bengio, "Word representation: A simple and general method for semi-supervised learning," in *Proceedings of the Association for Computational Linguistics (ACL)*, Uppsala, Sweden, 2010, pp. 384–394.

[104] M. Xiao and Y. Guo, "Domain adaptation for sequence labeling tasks with a probabilistic language adaptation model," in *Proceedings of the International Conference on Machine Learning (ICML)*, 2013, pp. 293–301.

[105] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," in *Proceedings of International Conference on Learning Representations*, 2013.

[106] M. U. Gutmann and A. Hyvärinen, "Noise-contrastive estimation of unnormalized statistical models, with applications to natural image statistics," *The Journal of Machine Learning Research*, vol. 13, no. 1, pp. 307–361, 2012.

[107] Y. Bengio, A. Courville, and P. Vincent, "Representation learning: A review and new perspectives," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 8, pp. 1798–1828, 2013.

[108] M. Joshi, M. Dredze, W. W. Cohen, and C. P. Rosé, "What's in a domain? multi-domain learning for multi-attribute data," in *Proceedings of the North American Chapter of the Association for Computational Linguistics (NAACL)*, Atlanta, GA, 2013, pp. 685–690.

[109] S. Petrov and R. McDonald, "Overview of the 2012 shared task on parsing the web," in *Notes of the First Workshop on Syntactic Analysis of Non-Canonical Language (SANCL)*, vol. 59, 2012.

[110] T. Schnabel and H. Schütze, "Flors: Fast and simple domain adaptation for part-of-speech tagging," *Transactions of the Association of Computational Linguistics (TACL)*, vol. 2, pp. 51–62, 1 2014.

[111] A. Ratnaparkhi, "A maximum entropy model for part-of-speech tagging," in *Proceedings of Empirical Methods for Natural Language Processing (EMNLP)*, 1996, pp. 133–142.

[112] C. D. Santos and B. Zadrozny, "Learning character-level representations for part-of-speech tagging," in *Proceedings of the International Conference on Machine Learning (ICML)*, 2014, pp. 1818–1826.

[113] Y. Yang and J. Eisenstein, "Fast easy unsupervised domain adaptation with marginalized structured dropout," in *Proceedings of the Association for Computational Linguistics (ACL)*, Baltimore, MD, 2014.

[114] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Advances in Neural Information Processing Systems (NIPS)*, Lake Tahoe, 2013, pp. 3111–3119.

[115] E. Hovy, M. Marcus, M. Palmer, L. Ramshaw, and R. Weischedel, "Ontonotes: The 90% solution," in *Proceedings of the North American Chapter of the Association for Computational Linguistics (NAACL)*, New York, NY, 2006, pp. 57–60.

[116] K. Toutanova, D. Klein, C. D. Manning, and Y. Singer, "Feature-rich part-of-speech tagging with a cyclic dependency network," in *Proceedings of the North American Chapter of the Association for Computational Linguistics (NAACL)*, 2003.

[117] J. Giménez and L. Marquez, "Svmtool: A general POS tagger generator based on support vector machines," in *Proceedings of the Language Resources and Evaluation Conference*, 2004.

[118] X.-H. Phan, *Crftagger: crf english pos tagger*, `http://crftagger.sourceforge.net`, 2006.

[119] A. Kroch and A. Taylor, *Penn-helsinki parsed corpus of middle english, second edition*, `http://www.ling.upenn.edu/hist-corpora/PPCME2-RELEASE-3/index.html`, 2000.

[120] T. Moon and J. Baldridge, "Part-of-speech tagging for middle english through alignment and projection of parallel diachronic texts.," in *Proceedings of Empirical Methods for Natural Language Processing (EMNLP)*, 2007, pp. 390–399.

[121] M. Rissanen, M. Kytö, and M. Palander-Collin, *Early English in the computer age: Explorations through the Helsinki Corpus*, 11. Walter de Gruyter, 1993.

[122] C. D. Manning, M. Surdeanu, J. Bauer, J. Finkel, S. J. Bethard, and D. McClosky, "The Stanford CoreNLP natural language processing toolkit," in *Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, 2014, pp. 55–60.

[123] P. F. Brown, P. V. Desouza, R. L. Mercer, V. J. D. Pietra, and J. C. Lai, "Class-based n-gram models of natural language," *Computational linguistics*, vol. 18, no. 4, pp. 467–479, 1992.

[124] G. Schneider, H. M. Lehmann, and P. Schneider, "Parsing early and late modern english corpora," *Literary and Linguistic Computing*, 2014.

[125] W. Ling, C. Dyer, A. Black, and I. Trancoso, "Two/too simple adaptations of word2vec for syntax problems," in *Proceedings of the North American Chapter of the Association for Computational Linguistics (NAACL)*, Denver, CO, 2015.

[126] H. Daumé III, "Frustratingly easy domain adaptation," in *Proceedings of the Association for Computational Linguistics (ACL)*, Prague, 2007.

[127] M. Dredze, A. Kulesza, and K. Crammer, "Multi-domain learning by confidence-weighted parameter combination," *Machine Learning*, vol. 79, no. 1-2, pp. 123–149, 2010.

[128] D. Wang, C. Xiong, and W. Y. Wang, "Automatic domain partitioning for multi-domain learning.," in *Proceedings of Empirical Methods for Natural Language Processing (EMNLP)*, Seattle, WA, 2013, pp. 869–873.

[129] P. F. Brown, V. J. Della Pietra, P. V. deSouza, J. C. Lai, and R. L. Mercer, "Class-Based N-Gram models of natural language," *Computational Linguistics*, vol. 18, pp. 18–4, 1990.

[130] R. Collobert and J. Weston, "A unified architecture for natural language processing: Deep neural networks with multitask learning," in *Proceedings of the International Conference on Machine Learning (ICML)*, 2008, pp. 160–167.

[131] A. Mnih and G. E. Hinton, "A scalable hierarchical distributed language model," in *Advances in Neural Information Processing Systems (NIPS)*, 2009, pp. 1081–1088.

[132] P. Dhillon, D. P. Foster, and L. H. Ungar, "Multi-view learning of word embeddings via cca," in *Advances in Neural Information Processing Systems (NIPS)*, Granada, 2011, pp. 199–207.

[133] D. Bamman, C. Dyer, and N. A. Smith, "Distributed representations of geographically situated language," in *Proceedings of the Association for Computational Linguistics (ACL)*, Baltimore, MD, 2014, pp. 828–834.

[134] S. Scheible, R. J. Whitt, M. Durrell, and P. Bennett, "Evaluating an 'off-the-shelf' POS-tagger on early modern German text," in *Proceedings of the 5th ACL-HLT workshop on language technology for cultural heritage, social sciences, and humanities*, 2011, pp. 19–23.

[135] M. Mathioudakis and N. Koudas, "Twittermonitor: Trend detection over the twitter stream," in *Proceedings of the ACM SIGMOD International Conference on Management of data (SIGMOD)*, 2010, pp. 1155–1158.

[136] W.-t. Yih, M.-W. Chang, X. He, and J. Gao, "Semantic parsing via staged query graph generation: Question answering with knowledge base," in *Proceedings of the Association for Computational Linguistics (ACL)*, Beijing, China, 2015.

[137] J. Tang, M. Qu, M. Wang, M. Zhang, J. Yan, and Q. Mei, "Line: Large-scale information network embedding," in *Proceedings of the Conference on World-Wide Web (WWW)*, 2015.

[138] S. Guo, M.-W. Chang, and E. Kiciman, "To link or not to link? a study on end-to-end tweet entity linking," in *Proceedings of the North American Chapter of the Association for Computational Linguistics (NAACL)*, Atlanta, GA, 2013.

[139] Y. Yang and M.-W. Chang, "S-mart: Novel tree-based structured learning algorithms applied to tweet entity linking," in *Proceedings of the Association for Computational Linguistics (ACL)*, Beijing, China, 2015.

[140] A. E. Cano, G. Rizzo, A. Varga, M. Rowe, M. Stankovic, and A.-S. Dadzie, "Making sense of microposts (# microposts2014) named entity extraction & linking challenge," *Making Sense of Microposts (# Microposts2014)*, 2014.

[141] Y. Fang and M.-W. Chang, "Entity linking on microblogs with spatial and temporal signals," *Transactions of the Association for Computational Linguistics (TACL)*, 2014.

[142] H. Kwak, C. Lee, H. Park, and S. Moon, "What is Twitter, a social network or a news media?" In *Proceedings of the Conference on World-Wide Web (WWW)*, 2010, pp. 591–600.

[143] S. Wu, J. M. Hofman, W. A. Mason, and D. J. Watts, "Who says what to whom on twitter," in *Proceedings of the Conference on World-Wide Web (WWW)*, 2011, pp. 705–714.

[144] R. Socher, D. Chen, C. D. Manning, and A. Y. Ng, "Reasoning with neural tensor networks for knowledge base completion," in *Advances in Neural Information Processing Systems (NIPS)*, Lake Tahoe, 2013.

[145] B. Yang, W.-t. Yih, X. He, J. Gao, and L. Deng, "Embedding entities and relations for learning and inference in knowledge bases," in *Proceedings of the International Conference on Learning Representations (ICLR)*, 2015.

[146] D. Carmel, M.-W. Chang, E. Gabrilovich, B.-J. P. Hsu, and K. Wang, "Erd'14: Entity recognition and disambiguation challenge," in *ACM SIGIR Forum*, 2014, pp. 63–77.

[147] D. M. Blei, A. Y. Ng, and M. I. Jordan, "Latent dirichlet allocation," *The Journal of machine Learning research*, vol. 3, pp. 993–1022, 2003.

[148] J. L. Fischer, "Social influences on the choice of a linguistic variant," *Word*, vol. 14, pp. 47–56, 1958.

[149] W. Labov, "The social motivation of a sound change," *Word*, vol. 19, no. 3, pp. 273–309, 1963.

[150] P. Nakov, Z. Kozareva, A. Ritter, S. Rosenthal, V. Stoyanov, and T. Wilson, "Semeval-2013 task 2: Sentiment analysis in twitter," in *Proceedings of the 7th International Workshop on Semantic Evaluation, SemEval*, 2013.

[151] J. Tang, H. Gao, and H. Liu, "Mtrust: Discerning multi-faceted trust in a connected world," in *Proceedings of the fifth ACM international conference on Web search and data mining*, ACM, 2012, pp. 93–102.

[152] S. Rosenthal, P. Nakov, S. Kiritchenko, S. M. Mohammad, A. Ritter, and V. Stoyanov, "Semeval-2015 task 10: Sentiment analysis in twitter," in *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval)*, 2015.

[153] D. Tang, F. Wei, B. Qin, M. Zhou, and T. Liu, "Building large-scale twitter-specific sentiment lexicon: A representation learning approach.," in *Proceedings of the 26th International Conference on Computational Linguistics (COLING)*, 2014, pp. 172–182.

[154] M. E. Newman, "The structure and function of complex networks," *SIAM review*, vol. 45, no. 2, pp. 167–256, 2003.

[155] A. A. Hagberg, D. A. Schult, and P Swart, "Exploring network structure, dynamics, and function using networkx," in *Proceedings of the 7th Python in Science Conferences (SciPy 2008)*, vol. 2008, 2008.

[156] R. A. Jacobs, M. I. Jordan, S. J. Nowlan, and G. E. Hinton, "Adaptive mixtures of local experts," *Neural computation*, vol. 3, no. 1, pp. 79–87, 1991.

[157] A. Bordes, N. Usunier, A. Garcia-Duran, J. Weston, and O. Yakhnenko, "Translating embeddings for modeling multi-relational data," in *Advances in Neural Information Processing Systems*, 2013.

[158] Z. Wang, J. Zhang, J. Feng, and Z. Chen, "Knowledge graph embedding by translating on hyperplanes.," in *Proceedings of the National Conference on Artificial Intelligence (AAAI)*, 2014.

[159] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel, "Backpropagation applied to handwritten zip code recognition," *Neural computation*, vol. 1, no. 4, pp. 541–551, 1989.

[160] N. Kalchbrenner, E. Grefenstette, and P. Blunsom, "A convolutional neural network for modelling sentences," in *Proceedings of the Association for Computational Linguistics (ACL)*, Baltimore, MD, 2014.

[161] Y. Kim, "Convolutional neural networks for sentence classification," in *Proceedings of Empirical Methods for Natural Language Processing (EMNLP)*, 2014, pp. 1746–1751.

[162] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," *ArXiv preprint arXiv:1412.6980*, 2014.

[163] J. Jiang and C. Zhai, "Instance weighting for domain adaptation in nlp," in *Proceedings of the Association for Computational Linguistics (ACL)*, Prague, 2007.

[164] R. F. Astudillo, S. Amir, W. Lin, M. Silva, and I. Trancoso, "Learning word representations from scarce and noisy data with embedding sub-spaces," in *Proceedings of the Association for Computational Linguistics (ACL)*, Beijing, China, 2015.

[165] M. Hagen, M. Potthast, M. Büchner, and B. Stein, "Webis: An ensemble for twitter sentiment detection," in *Proceedings of the 9th International Workshop on Semantic Evaluation*, 2015.

[166] A. Severyn and A. Moschitti, "Unitn: Training deep convolutional neural network for twitter sentiment classification," in *Proceedings of the 9th International Workshop on Semantic Evaluation*, 2015.

[167] H. Hamdan, P. Bellot, and F. Bechet, "Lsislif: Feature extraction and label weighting for sentiment analysis in twitter," in *Proceedings of the 9th International Workshop on Semantic Evaluation*, 2015.

[168] M. Al Boni, K. Q. Zhou, H. Wang, and M. S. Gerber, "Model adaptation for personalized opinion analysis," in *Proceedings of the Association for Computational Linguistics (ACL)*, Beijing, China, 2015.

[169] R. C Bunescu and M. Pasca, "Using encyclopedic knowledge for named entity disambiguation.," in *Proceedings of the European Chapter of the Association for Computational Linguistics (EACL)*, 2006.

[170] S. Cucerzan, "Large-scale named entity disambiguation based on wikipedia data.," in *Proceedings of Empirical Methods for Natural Language Processing (EMNLP)*, 2007.

[171] D. Milne and I. H. Witten, "Learning to link with Wikipedia," in *Proceedings of the International Conference on Information and Knowledge Management (CIKM)*, 2008.

[172] H. Huang, Y. Cao, X. Huang, H. Ji, and C.-Y. Lin, "Collective tweet wikification based on semi-supervised graph regularization.," in *Proceedings of the Association for Computational Linguistics (ACL)*, Baltimore, MD, 2014.

[173] Y. Guo, B. Qin, T. Liu, and S. Li, "Microblog entity linking by leveraging extra posts," in *Proceedings of Empirical Methods for Natural Language Processing (EMNLP)*, Seattle, WA, 2013.

[174] W. Shen, J. Wang, P. Luo, and M. Wang, "Linking named entities in tweets with knowledge base via user interest modeling," in *Proceedings of Knowledge Discovery and Data Mining (KDD)*, 2013.

[175] M. Collins, "Discriminative training methods for hidden markov models: Theory and experiments with perceptron algorithms," in *Proceedings of Empirical Methods for Natural Language Processing (EMNLP)*, 2002, pp. 1–8.

[176] B. Taskar, C. Guestrin, and D. Koller, "Max-margin markov networks," in *NIPS*, 2003.

[177] I. Tsochantaridis, T. Hofmann, T. Joachims, and Y. Altun, "Support vector machine learning for interdependent and structured output spaces," in *ICML*, 2004.

[178] D. Weiss, C. Alberti, M. Collins, and S. Petrov, "Structured training for neural network transition-based parsing," in *Proceedings of the Association for Computational Linguistics (ACL)*, Beijing, China, 2015.

[179] M. Speriosu, N. Sudan, S. Upadhyay, and J. Baldridge, "Twitter polarity classification with label propagation over lexical links and the follower graph," in *Proceedings of Empirical Methods for Natural Language Processing (EMNLP)*, Edinburgh, UK, 2011, pp. 53–63.

[180] T. G. Dietterich, "Ensemble methods in machine learning," in *International workshop on multiple classifier systems*, Springer, 2000, pp. 1–15.

[181] L. Breiman, "Bagging predictors," *Machine learning*, vol. 24, no. 2, pp. 123–140, 1996.

[182] ——, "Random forests," *Machine learning*, vol. 45, no. 1, pp. 5–32, 2001.

[183] Y. Freund, R. E. Schapire, *et al.*, "Experiments with a new boosting algorithm," in *ICML*, vol. 96, 1996, pp. 148–156.